PART II

ARCHITECTURES AND THEIR MATHEMATICAL LOGIC

Inductive Biases and Symmetry Groups • The Transformer Family
Flash Attention, RoPE, Linear Attention, KV-Cache
Sequential Architectures: LSTM → HiPPO/S4 → Mamba/RWKV • GNN and the WL Test

Eight-Part Monograph Series on the Mathematical Principles of Neural Networks
Independent of Companion Volumes — Complete in Itself

Silvano Salvador

© Mathematics of Deep Learning Series

Mathematics of Deep Learning — Part I: The Geometric Foundations

Preface: On the Mathematical Logic of Architecture Design

Architecture design is not engineering intuition dressed in mathematical clothing. The choices that distinguish a Transformer from an LSTM, a graph neural network from a convolutional network, a selective state space model from a linear recurrent system — each of these choices encodes a precise mathematical prior about the structure of the data and the tasks the network will be asked to perform. This prior is called an inductive bias, and the mathematical language in which it is most precisely expressed is the language of group theory, symmetry, and representation theory.

This volume pursues that language with full rigour. We begin with the general theory of inductive biases and their algebraic formulation as equivariance constraints. We then apply this framework systematically to the Transformer family, tracing how each architectural innovation — scaled dot-product attention, multi-head attention, positional encodings, Flash Attention, RoPE, linear attention, KV-Cache — modifies the mathematical structure of the computation and its associated biases and efficiency properties. We then turn to sequential architectures: the LSTM (a learnable linear system with gated

nonlinearity), the HiPPO framework (optimal polynomial projection of input history), the S4 model (a linear recurrence with structured transitions), Mamba (content-selective SSMs), and RWKV (linear attention with decaying memory). Finally, we treat graph neural networks through the lens of the Weisfeiler-Lehman hierarchy of graph isomorphism tests, providing the sharpest known characterisation of their expressive power.

A recurring theme will be the tradeoff triangle: expressivity, computational efficiency, and sample efficiency. Every architectural choice moves a model along this triangle, and understanding the mathematical mechanism of each choice is the key to principled architecture design. We make this tradeoff explicit at every step.

Chapter 1: Inductive Biases, Symmetry Groups, and Equivariant Maps

An inductive bias is any constraint on the hypothesis class that is not derived from data. Every neural network architecture embodies inductive biases, whether or not they are consciously chosen. A fully connected network with unconstrained weight matrices has a relatively weak inductive bias — in principle it can represent any function, but this very generality means it must learn from scratch the regularity that domain-specific architectures build in. The question is not whether an architecture has inductive biases, but which biases it has and whether those biases align with the true regularity of the problem.

The most principled source of architectural inductive bias is symmetry. If the target function f: X → Y is known to respect a symmetry group G acting on X (and possibly Y), then restricting the hypothesis class to G-equivariant functions both reduces sample complexity (fewer parameters need to be learned) and improves generalisation (the learned function cannot violate the symmetry on new inputs). The mathematical formalisation of this idea is equivariance.

1.1 Group Actions and Equivariance

Let G be a group acting on spaces X and Y via group actions ϱ_X: G × X → X and ϱ_Y: G × Y → Y. A function f: X → Y is G-equivariant if:

$$f(\rho_X(g, x)) = \rho_Y(g, f(x)) \quad \text{for all } g \in G, x \in X$$

Intuitively: if we transform the input by g, the output transforms by g in the same way. The special case where ϱ_Y is the trivial action (g • y = y for all g) gives G-invariance: f(ϱ_X(g, x)) = f(x). Invariance is a stronger constraint than equivariance — invariant functions ignore the group action entirely, while equivariant functions track it in a structured way.

Examples of symmetry groups in deep learning.

- Translation group (ℝ^d, +): acting on images by pixel shifts. Convolution is translation-equivariant: if the image is shifted by t, the feature map shifts by t. This is the foundational bias of CNNs.
- Permutation group S_n: acting on sets of n elements by relabelling. A set-valued function (e.g., classifying a point cloud) should be permutation-invariant. PointNet achieves this via symmetric aggregation (max-pooling or sum-pooling over point features).
- Rotation group SO(3) and rigid motion group SE(3): acting on 3D molecular structures. Equivariant graph networks (e-GNN, SE(3)-Transformer, SEGNN) build in these symmetries for protein structure prediction and molecular property regression.
- Lorentz group SO(1,3): acting on 4-vectors in particle physics. LorentzNet builds exact Lorentz-equivariance for particle physics event classification.

Theorem 1.1 (Sample Complexity Reduction from Equivariance)

Let H be a hypothesis class of all continuous functions $X \to Y$ with Lipschitz constant L, and let $H_G \subset H$ be the subclass of G-equivariant functions. If G has measure $\mu(G)$ under a normalised Haar measure, then the Rademacher complexity satisfies:

$\mathfrak{R}_m(H_G) \leq \mathfrak{R}_m(H) / \sqrt{|G|}$ *(for finite G, $|G|$ = group order)*

This bound implies that building G-equivariance into the architecture is equivalent to augmenting the training set by a factor of $|G|$ in terms of its effect on generalisation bounds. For the permutation group S_n (order $n!$), this is an exponential reduction. For the translation group acting on $W \times H$ images (roughly WH translations), this corresponds to the well-known fact that data augmentation with translations is far less efficient than a convolutional architecture that builds translation equivariance in exactly.

1.2 Representation Theory and the Structure of Equivariant Linear Layers

The set of all G-equivariant linear maps between representation spaces is itself a vector space, and its structure is completely determined by the

representation theory of G. This is the key result that makes equivariant network design systematic rather than ad hoc.

Let ϱ: G → GL(V) and τ: G → GL(W) be linear representations of G on finite-dimensional vector spaces V and W over ℝ. A linear map T: V → W is equivariant (an intertwiner) if Tϱ(g) = τ(g)T for all g ∈ G. By Schur's Lemma:

Theorem 1.2 (Schur's Lemma — Structure of Equivariant Maps)

(i) If ρ and τ are irreducible representations (irreps) over ↛ then every equivariant map T: V ↛ W is either zero (if ρ ≉ τ) or a scalar multiple of the canonical isomorphism (if ρ ≅ τ).

(ii) More generally, by the decomposition of ρ and τ into irreps ρ = ⊕_k m_k ρ_k and τ = ⊕_k n_k ρ_k, the space of equivariant maps decomposes as:

Hom_G(V, W) ≅ ⊕_k 𝕽^{n_k × m_k}

so the number of free parameters in an equivariant linear layer is Σ_k n_k m_k, compared to (dim V) (dim W) for an unconstrained linear map. The multiplicities m_k, n_k count how many times each irrep appears in the input and output representations. This decomposition is the mathematical basis for equivariant network design: represent the data as a direct sum of irreps, and parametrise layers as block-diagonal in the irrep basis with learnable multiplicity-mixing matrices.

Steerable CNNs. Cohen and Welling's Steerable CNNs implement this programme for the 2D rotation group SO(2) and the dihedral groups D_n. The feature maps at each layer are G-steereable vector bundles: at each spatial location, the feature vector transforms as a direct sum of irreducible representations of the local symmetry group. The learnable filter banks are constrained to be equivariant (their weights lie in the spaces Hom_G(V_in, V_out) determined by Schur's Lemma), and the spatial convolution provides the global translation equivariance. The result is a network that is exactly equivariant to the full group G = ℝ^2 ⋊ SO(2) of planar Euclidean motions.

1.3 The Universal Approximation Theorem for Equivariant Functions

A natural question is whether imposing equivariance constraints limits the class of functions a network can approximate. The answer is negative: the space of continuous G-equivariant functions is dense in the space of all G-equivariant functions under mild conditions, and equivariant networks are universal approximators within this restricted but natural class.

Theorem 1.3 (Equivariant Universal Approximation)

Let G be a compact group acting continuously on compact Hausdorff spaces X and Y. Let H_G be the space of continuous G-equivariant functions X → Y with the uniform norm. Then the class of equivariant neural networks with sufficient width and depth is dense in H_G.

The key steps in the proof are: (1) The equivariant Peter-Weyl theorem decomposes H_G into irreducible components indexed by the dual Ĝ of G; (2) each irreducible component is approximated by equivariant polynomials (via the Stone-Weierstrass theorem for G-algebras); (3) equivariant polynomials are expressed as products of equivariant linear maps and pointwise nonlinearities. The theorem implies that no expressivity is lost by imposing equivariance: equivariant networks can represent any continuous equivariant function, and only continuous equivariant functions.

Chapter 2: The Transformer Family

The Transformer architecture (Vaswani et al., 2017) is the dominant architecture in contemporary deep learning. Its mathematical core — scaled dot-product attention — is a kernel method: it computes a weighted combination of value vectors, where weights are determined by a kernel function (the dot product) evaluated between query and key vectors. Understanding the Transformer at this level of mathematical depth reveals why it is expressive, why it scales, and why each subsequent innovation (Flash Attention, RoPE, linear attention, KV-Cache) addresses a specific mathematical limitation of the original design.

2.1 Scaled Dot-Product Attention: Exact Formulation

Let X ∈ ℝ^{L×d} be a sequence of L token embeddings of dimension d. For a single attention head, the queries, keys, and values are obtained by linear projections:

$$Q = X\,W_Q \in \mathbb{R}^{L\times d_k},\quad K = X\,W_K \in \mathbb{R}^{L\times d_k},\quad V = X\,W_V \in \mathbb{R}^{L\times d_v}$$

where W_Q, W_K ∈ ℝ^{d×d_k} and W_V ∈ ℝ^{d×d_v} are the projection matrices. The attention output is:

$$\mathrm{Attn}(Q, K, V) = \mathrm{softmax}(QK^T / \sqrt{d_k})\,V \in \mathbb{R}^{L\times d_v}$$

The attention weight matrix A = softmax(QK^T / √d_k) ∈ ℝ^{L×L} has entry A_{ij} = exp(q_i^T k_j / √d_k) / Σ$_k$ exp(q_i^T k_$_k$ / √d_k), representing the normalised affinity between position i (as a query) and position j (as a key). The output at position i is the A_{ij}-weighted average of value vectors: out_i = Σ_j A_{ij} v_j.

The √d_k scaling. If q_i and k_j are independently sampled from ℕ(0, I_{d_k}), their dot product q_i^T k_j has variance d_k. Without scaling, the pre-softmax logits grow as √d_k in standard deviation, pushing the softmax into saturation (near-zero gradients everywhere except at the argmax). Dividing by √d_k normalises the variance to 1, keeping the softmax in its gradient-informative

regime throughout training. This scaling is an instance of a general principle: the input to a softmax should be O(1) in magnitude for well-conditioned gradients.

Multi-head attention. Multi-head attention runs h parallel attention heads with independent projections (W_Q^{(i)}, W_K^{(i)}, W_V^{(i)}) and concatenates their outputs:

$$MHA(X) = Concat(head_1, \ldots, head_h)\, W_O, \quad head_i = Attn(X\, W_Q^{(i)}, X\, W_K^{(i)}, X\, W_V^{(i)})$$

where W_O ∈ ℝ^{hd_v × d} projects the concatenated heads back to the model dimension. Typically d_k = d_v = d/h so the total parameter count is $O(d^2)$ per attention layer regardless of the number of heads. Each head can specialise in a different type of similarity (syntactic, semantic, positional), and the multi-head structure provides an ensemble of kernel machines operating simultaneously.

Theorem 2.1 (Attention as a Universal Approximator of Sequence-to-Sequence Maps)

A single-layer Transformer with sufficient number of attention heads and embedding dimension can approximate any continuous sequence-to-sequence function f: (ℝ^d)^L → (ℝ^d)^L on compact domains to arbitrary precision. The proof proceeds by showing that multi-head attention can implement any weighted sum over the input sequence, and that the feed-forward sublayer can then apply an arbitrary continuous function to each position independently. Combining the two sublayers, any pointwise-continuous permutation-equivariant function is reachable. With positional encodings breaking the permutation equivariance, any continuous function on fixed-length sequences is approximable.

Depth (multiple layers) is needed for functions that require multi-round reasoning: each layer can refine the representation based on the globally attended context from the previous layer. Empirically, depth is more parameter-efficient than width for complex compositional tasks.

2.2 Positional Encodings: From Additive Sinusoids to RoPE

The attention mechanism as defined above is permutation-equivariant: any permutation of the input sequence produces a permuted output. This is problematic for tasks where order matters (language modelling, time series

forecasting, machine translation). Positional encodings break this symmetry by injecting position-dependent information into the input representations.

Sinusoidal positional encodings (Vaswani et al., 2017). The original Transformer adds to each input embedding x_t a fixed positional vector PE(t) with components:

$$PE(t, 2k) = sin(t / 10000^{2k/d}), \quad PE(t, 2k+1) = cos(t / 10000^{2k/d})$$

The choice of sinusoids is motivated by the observation that PE(t + Δt) is a linear function of PE(t): there exists a rotation matrix R_{Δt} such that PE(t + Δt) = R_{Δt} PE(t). This means the attention mechanism can in principle learn to attend to tokens at a fixed relative offset Δt by learning to apply R_{Δt} to the queries or keys. However, the linearity holds for the raw positional vectors, not for the projected queries and keys — the projections W_Q and W_K distort the linear relationship, making relative position information harder to exploit than the design intention suggests.

Rotary Position Embedding (RoPE, Su et al., 2021). RoPE addresses this by encoding position directly in the dot product between query and key, rather than in the input embeddings. The key insight is to rotate the query and key vectors for position t by a rotation matrix Φ_t:

$$\alpha(q_t, k_s) = (R_t\, q_t)^T (R_s\, k_s) = q_t^T R_t^T R_s\, k_s = q_t^T R_{s-t}\, k_s$$

where R_t is a block-diagonal rotation matrix with 2×2 rotation blocks for each pair of dimensions, rotated by angle tθ_k = t × 10000^{-2k/d} for the k-th dimension pair. The resulting attention score depends only on the relative position s - t, not on the absolute positions s and t separately. This is the relative position equivariance property that the original sinusoidal encoding could not achieve exactly.

Theorem 2.2 (RoPE Relative Position Equivariance)

Let $R_t \in SO(d)$ be the RoPE rotation at position t, defined as the block-diagonal matrix with 2×2 rotation blocks $R(\theta_k t)$ for k = 1,...,d/2. Then for any query q and key k:

$(R_t\, q)^T (R_s\, k) = q^T R_{s-t}\, k$

The proof follows from the identity $R_t^T R_s = R_{s-t}$ (rotation by $-(t\theta_k)$ followed by rotation by $s\theta_k$ equals rotation by $(s-t)\theta_k$ for each block). This means the dot-product attention score with RoPE depends only on the relative displacement $s-t$ between positions, making the attention pattern translation-equivariant. RoPE also extrapolates better than additive sinusoidal encodings to sequence lengths beyond those seen during training because the relative rotation angle $(s-t)\theta_k$ grows linearly in the displacement, maintaining consistent angular scale at any sequence length.

ALiBi and other positional schemes. ALiBi (Press et al., 2022) takes a simpler approach: instead of modifying the queries and keys, it adds a head-specific linear bias $-m_h |s - t|$ to the pre-softmax attention score at head h, where m_h is a head-specific slope. This penalises long-range attention linearly in distance, implementing a soft local attention bias. ALiBi is parameter-free (m_h is fixed, not learned), has excellent length extrapolation properties (the linear penalty smoothly extends to unseen distances), and has become the positional encoding of choice for several production language models (BLOOM, MPT). Its simplicity, however, comes at the cost of expressivity: it cannot represent arbitrary relative position functions, only linear-decay functions.

2.3 Flash Attention: IO-Aware Exact Attention

The standard attention computation requires materialising the L×L attention weight matrix A, which costs $O(L^2)$ memory. For L = 100,000 (typical for long-context models), this is 10^{10} entries — 40 GB in float32. Flash Attention (Dao et al., 2022; Dao, 2023) reduces this to O(L) memory while computing exactly the same output as standard attention, by exploiting the hierarchical memory structure of GPU hardware.

The memory bottleneck. Modern GPUs have a fast on-chip SRAM (typically 20–40 MB for an A100) and a large off-chip HBM (40–320 GB). The bandwidth to SRAM is ~19 TB/s but its capacity is tiny; the bandwidth to HBM is ~2 TB/s. Standard attention requires multiple HBM reads: read Q, K to compute QK^T (write to HBM); read QK^T, apply softmax (write to HBM); read A, V to

compute AV (write output). Each read/write is an HBM access, and HBM bandwidth is the bottleneck.

Tiled computation. Flash Attention processes the sequence in tiles of size B_r × B_c (block rows × block cols), where B_r and B_c are chosen so that tiles of Q, K, V fit in SRAM simultaneously. For each tile of queries (rows), it iterates over all tiles of keys and values (cols), accumulating the softmax-weighted value sum in SRAM without ever writing the full L×L attention matrix to HBM. The softmax normalisation across tiles requires the online softmax algorithm:

Algorithm 2.1 (Online Softmax for Flash Attention)

To compute softmax(x_1, ..., x_L) without storing all x_i simultaneously, maintain running statistics (m, d) where m = max seen so far and d = sum of exp(x_i - m) seen so far:

For each new x_{i+1}: m_new = max(m, x_{i+1}); d_new = d × exp(m - m_new) + exp(x_{i+1} - m_new)

The correction factor exp(m - m_new) rescales the previous sum to the new max. At the end, softmax_i = exp(x_i - m) / d. This algorithm is numerically stable (equivalent to standard softmax after subtracting the global max) and requires O(1) memory per streaming pass.

Flash Attention extends this to matrix blocks: for a query block Q_i and key-value blocks (K_j, V_j), maintain per-row running max m_i^{(j)} and sum d_i^{(j)}, and accumulate the partial attention output O_i^{(j)} = A_{ij}V_j (where A_{ij} = exp(Q_i K_j^T/√d_k - m_i^{(j)})). After processing all key blocks, the final output O_i = Σ_j O_i^{(j)} × (d_i^{(j)}/d_i^{(final)}) is exact.

Complexity analysis. Standard attention: O(L² d) FLOPs, O(L² + Ld) memory, O(L²/B) HBM reads. Flash Attention: O(L² d) FLOPs (same!), O(Ld) memory (reduced from O(L²)), O(L² d / M) HBM reads (where M is SRAM size). The FLOP count is unchanged — Flash Attention does not reduce the arithmetic work. Its advantage is entirely in HBM bandwidth, which is the true bottleneck for L ≥ 1024 on modern GPUs. Empirically, Flash Attention achieves 2–4× wall-clock speedup over standard attention at L = 2048 and 5–10× at L = 4096 on A100 GPUs.

Version	Memory	FLOPs	Backward Pass	Key Feature

Standard Attention	$O(L^2)$	$O(L^2d)$	Standard autodiff	Simple but memory-bound
Flash Attention v1	$O(Ld)$	$O(L^2d)$	Recompute A in backward	2-4× speedup at L=2048
Flash Attention v2	$O(Ld)$	$O(L^2d)$	Optimised backward	Better GPU utilisation
Flash Attention v3	$O(Ld)$	$O(L^2d)$	Async overlapped	Optimised for H100/Hopper
Flash Decoding	$O(Ld)$	$O(Ld)$	N/A (inference)	Parallelises KV-Cache lookup

2.4 KV-Cache: The Memory-Bandwidth Tradeoff at Inference

At inference time, transformer language models generate tokens autoregressively: given tokens (t_1, ..., t_n), they predict t_{n+1}, then given (t_1, ..., t_{n+1}) they predict t_{n+2}, and so on. Naively, each forward pass recomputes the keys and values for all previous tokens, costing $O(nL)$ per token (n is the current position, L the context length). The KV-Cache eliminates this redundancy by caching the computed key and value matrices from previous steps.

Mechanism. At step n, the keys and values for the new token t_n are computed and appended to the cache: K_cache ← [K_cache; k_n^T], V_cache ← [V_cache; v_n^T]. The attention at step n then reads: Q_n = q_n (1×d_k), K_cache (n×d_k), V_cache (n×d_v), and computes the attention output using the cached KV without recomputation. The cost per token drops from $O(n^2 d)$ (naive) to $O(nd)$ (with cache) — a factor of n reduction in compute.

Memory cost. The KV-Cache stores 2 × n_layers × n_heads × n × d_head values per sequence in the batch. For a typical large model (96 layers, 96 heads, d_head = 128), with n = 100,000 tokens and fp16 precision, this is 2 × 96 × 96 × 100,000 × 128 × 2 bytes ≈ 470 GB per sequence — exceeding the GPU memory of

any current hardware. This is the primary challenge of long-context inference and motivates a family of KV-Cache compression techniques:

- Quantisation: store KV in 4-bit or 8-bit formats (KVQuant, KIVI), accepting small approximation error for 4-8× memory reduction.
- Eviction: dynamically discard cache entries for low-attention-weight tokens (H2O, ScissorHands), keeping only the most attended tokens.
- Sparse attention: at inference, compute attention only over a sliding window or over selected anchor tokens (Sliding Window, StreamingLLM).
- Multi-Query Attention (MQA): share a single key-value head across all query heads. Cache size drops by factor h (number of heads). Used in Falcon, PaLM-2.
- Grouped-Query Attention (GQA): intermediate between MHA and MQA. Group query heads into G groups, with one KV head per group. Cache drops by factor h/G. Used in Llama-2 70B, Mistral, Gemma.

Theorem 2.3 (GQA Approximation Quality)

Let MHA be multi-head attention with h heads and GQA with G < h groups (h/G queries per group). GQA is equivalent to MHA with the constraint that all queries in group g share the same key and value projections W_K^{(g)}, W_V^{(g)}. The approximation error of GQA relative to MHA decreases as G increases: G = h recovers MHA exactly, G = 1 gives MQA. For any fixed target accuracy ε > 0 in terms of output Frobenius norm, there exists G = G*(h, ε, X) such that GQA with G ≥ G* groups achieves output error ≤ ε for input X. Empirically, G = h/8 (e.g., G = 8 for h = 64) recovers essentially all of MHA's performance at 8× reduced KV-Cache size.*

2.5 Linear Attention and the Kernel Interpretation

Standard softmax attention has $O(L^2)$ cost due to the L×L attention matrix. Linear attention replaces the softmax with a different, factorised attention

kernel that decomposes the L×L matrix into a product of L×d and d×L matrices, reducing cost to $O(Ld^2)$.

The key observation is that softmax attention can be written as:

$$out_i = \Sigma_j \exp(q_i^T k_j / \sqrt{d_k})\, v_j \,/\, \Sigma_j \exp(q_i^T k_j / \sqrt{d_k}) = \Sigma_j K(q_i, k_j)\, v_j \,/\, \Sigma_j K(q_i, k_j)$$

where $K(q, k) = \exp(q^T k / \sqrt{d_k})$ is a positive definite kernel. Linear attention replaces K with a kernel that factors: $K(q, k) = \varphi(q)^T \varphi(k)$ for some feature map $\varphi: \mathbb{R}^{d_k} \to \mathbb{R}^r$. Then:

$$out_i = \varphi(q_i)^T (\Sigma_j \varphi(k_j)\, v_j^T) \,/\, \varphi(q_i)^T (\Sigma_j \varphi(k_j))$$

The sum $S = \Sigma_j \varphi(k_j)\, v_j^T \in \mathbb{R}^{r \times d_v}$ and $z = \Sigma_j \varphi(k_j) \in \mathbb{R}^r$ can be computed incrementally: processing $j = 1, \ldots, L$ sequentially costs $O(Lrd_v)$ total. If $r = O(d_k)$, this is $O(Ld_k d_v) = O(Ld^2/h^2)$ — linear in L.

Feature maps and the exponential kernel. The challenge is choosing φ such that $\varphi(q)^T \varphi(k)$ approximates $\exp(q^T k / \sqrt{d_k})$ well while keeping r small. Random Fourier Features (Rahimi and Recht, 2007) approximate shift-invariant kernels by sampling Fourier components. For the exponential kernel, exact positive feature maps exist: Performer (Choromanski et al., 2021) uses orthogonal random features FAVOR+ based on:

$$\varphi(x) = \exp(-\|x\|^2/2) \times (\exp(\omega_1^T x), \ldots, \exp(\omega_r^T x)) \quad \omega_i \sim \mathcal{N}(0, I/\sqrt{d_k})$$

Retentive Networks and the recurrence-attention duality. RetNet (Sun et al., 2023) and related models (GLA, HGRN, RWKV) exploit the fact that linear attention with causal masking is equivalent to a linear recurrent neural network. Specifically, for causal linear attention with decay γ:

$$s_t = \gamma\, s_{t-1} + \varphi(k_t)\, v_t^T, \quad out_t = \varphi(q_t)^T s_t \,/\, normaliser$$

The hidden state $s_t \in \mathbb{R}^{r \times d_v}$ is the compressed summary of all past keys and values, updated by a simple outer product addition with exponential decay. During training, this can be parallelised using the chunk-wise form (compute attention within chunks in parallel, pass hidden states between chunks sequentially). During inference, it reduces to an O(1)-per-token

recurrence. This training-inference duality is the core advantage of linear attention over quadratic attention and of recurrent over attention-based models.

Chapter 3: Sequential Architectures — From LSTM to Mamba

The problem of learning from sequential data with long-range dependencies is one of the oldest challenges in machine learning. The progression from Hidden Markov Models through LSTMs to state space models is a mathematical story of increasingly powerful representations of memory, each with characteristic tradeoffs between the length of dependency, the cost of training, and the cost of inference. The past three years have produced a genuine architectural revolution in this space: selective state space models (Mamba) and hybrid architectures challenge the transformer's dominance on sequence modelling tasks while offering O(L) inference cost rather than $O(L^2)$.

3.1 Long Short-Term Memory: The Gating Solution to Vanishing Gradients

The LSTM (Hochreiter and Schmidhuber, 1997) was designed to solve the vanishing gradient problem for recurrent networks by introducing an explicit memory cell c_t with additive (rather than multiplicative) updates. Its gating mechanism is the archetype of all subsequent gated architectures.

The LSTM equations for hidden state h_{t-1} ∈ ℝ^h and cell state c_{t-1} ∈ ℝ^h, given input x_t ∈ ℝ^d:

$$i_t = \sigma(W_i\,[h_{t-1};\, x_t] + b_i) \quad \textit{[input gate]}$$

$$f_t = \sigma(W_f\,[h_{t-1};\, x_t] + b_f) \quad \textit{[forget gate]}$$

$$o_t = \sigma(W_o\,[h_{t-1};\, x_t] + b_o) \quad \textit{[output gate]}$$

$$g_t = \tanh(W_g\,[h_{t-1};\, x_t] + b_g) \quad \textit{[cell candidate]}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t$$

$$h_t = o_t \odot \tanh(c_t)$$

The cell state update c_t = f_t ⊙c_{t-1} + i_t ⊙g_t is additive when f_t ≈ 1 and i_t ≈ 0 (memorising), allowing gradients to flow back through many time steps

without decay. The forget gate f_t controls how much of the previous cell state is retained; if f_t ≈ 1 for many steps, the gradient ∂c_t/∂c_{t-k} = ∏τ f_τ ≈ 1, preventing vanishing. This is mathematically identical to the residual connection argument in Chapter 2 of Part I: an additive update prevents the product of Jacobians from decaying exponentially.

Theorem 3.1 (LSTM Gradient Flow)

For an LSTM, the gradient of the loss L at time T with respect to the cell state c_t is:

$$\partial L/\partial c_t = (\partial L/\partial c_T) \times \prod_{\tau=t+1}^{T} f_\tau + \text{lower-order terms}$$

If the forget gates $f_\tau \in (0, 1)$ for all τ, then the gradient decays as $\prod f_\tau < 1$ but polynomially (not exponentially) in (T-t) if f_τ is bounded away from 0. If $f_\tau = 1$ for all τ (open memory gate), the gradient is exactly preserved. This is in contrast to the vanilla RNN where the analogous product is $\prod_\tau \sigma'(z_\tau)\, \rho(W_h)$, which decays exponentially when the spectral radius $\rho(W_h) \times \|\sigma'\|_\infty < 1$.

3.2 The HiPPO Framework: Optimal Polynomial Memory

The HiPPO (High-order Polynomial Projection Operator) framework (Gu et al., 2020) provides a mathematically principled approach to the question: what is the optimal way for a recurrent system to remember its input history? The answer is to project the input history onto an orthogonal polynomial basis, and HiPPO derives the exact linear recurrence that implements this projection optimally.

Formal setup. Let u: [0, t] → ℝ be the input signal up to time t, and let ω(x, t) be a time-varying measure on [0, t] (e.g., uniform: ω(x, t) = 1/t). HiPPO seeks coefficients c(t) ∈ ℝ^N such that the N-dimensional polynomial approximation g(x, t) = Σ_{n=0}^{N-1} c_n(t) P_n(x) minimises the weighted L^2 error:

$$c(t) = \operatorname{argmin}_{c \in \mathbb{R}^N} \int_0^t (u(x) - g(x, t))^2\, \omega(x, t)\, dx$$

HiPPO derives that for any such measure ω, the optimal coefficients satisfy a linear ODE:

$$c'(t) = A(t)\, c(t) + B(t)\, u(t)$$

where A(t) and B(t) are explicitly computable from the measure ω and the chosen polynomial basis. This is a continuous-time linear recurrence, and it is the theoretically grounded generalisation of the LSTM cell state update to an arbitrary order of polynomial approximation.

Theorem 3.2 (HiPPO-LegS: Legendre Scaling Measure)

For the scaled Legendre measure ω(x, t) = 1/t × 𝟙{[0,t]}(x) (uniform over all history up to t), the HiPPO matrix A ∈ ℝ^{N×N} has entries:

A_{nk} = -(2n+1)^{1/2}(2k+1)^{1/2} if n > k

A_{nk} = -(n+1) if n = k

A_{nk} = 0 if n < k

and B_n = (2n+1)^{1/2}. The coefficients c_n(t) are the Legendre polynomial coefficients of the input history u(s), s ≤ t, under the uniform measure. This means the HiPPO state c(t) ∈ ℝ^N encodes the best N-th order polynomial approximation to the entire input history, with an approximation error that decays as O(1/N^2) in L². Critically, the A matrix is lower-triangular, making the recurrence causal and stable.

The significance of HiPPO is that it provides a principled justification for the specific structure of the recurrence matrices in S4 and its descendants. Rather than treating the recurrence matrix A as a free parameter to be learned from data (as in vanilla RNNs and LSTMs), HiPPO derives A analytically from the desired memory property. The result is a recurrence that can provably memorise input history of arbitrary length, limited only by the number of polynomial basis functions N.

3.3 S4: Structured State Spaces for Sequences

The S4 model (Gu et al., 2021) instantiates the HiPPO framework as a trainable neural network layer. A continuous-time state space model (SSM) is defined by the system:

$$x'(t) = Ax(t) + Bu(t), \quad y(t) = Cx(t) + Du(t)$$

where x(t) ∈ ℝ^N is the hidden state, u(t) ∈ ℝ is the scalar input, y(t) ∈ ℝ is the output, and A ∈ ℝ^{N×N}, B ∈ ℝ^{N×1}, C ∈ ℝ^{1×N}, D ∈ ℝ are the system matrices. Discretising with step size Δ using the bilinear (Tustin) method:

$$\bar{A} = (I - \Delta/2 \cdot A)^{-1} (I + \Delta/2 \cdot A), \quad \bar{\mathcal{B}} = (I - \Delta/2 \cdot A)^{-1} \Delta B$$

gives the discrete recurrence x_k = Ā x_{k-1} + B̄ u_k, y_k = C x_k + D u_k. Unrolling this recurrence gives a convolution y = K * u where K_k = C Ā^{k-1} B̄ is the SSM convolution kernel (also called the impulse response). Computing y via convolution costs O(L log L) using FFT, compared to O(L²) for naive recurrence unrolling.

Theorem 3.3 (SSM Convolution and Diagonalisation)

If A is diagonalisable as $A = V \Lambda V^{-1}$ with $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$, then:

$K_k = C \bar{A}^{k-1} \bar{\mathcal{B}} = (CV)\, \Delta\, \bar{A}^{k-1}\, (V^{-1}B) = \sum_{n=1}^{N} \alpha_n \bar{\lambda}_n^{k-1}$

where $\alpha_n = (CV)_n (V^{-1}B)_n$ and $\bar{\lambda}_n$ is the n-th eigenvalue of $\bar{A}$ (a Möbius transform of λ_n). The kernel K decomposes as a sum of N exponential sequences, one per eigenvalue. For stability, $|\bar{\lambda}_n| < 1$ (equivalently, $\mathrm{Re}(\lambda_n) < 0$). The diagonalisation reduces the N×N matrix power $\bar{A}^k$ to N scalar powers $\bar{\lambda}_n^k$, making the kernel computation O(NL) rather than $O(N^2 L)$.

S4 uses the NPLR (Normal Plus Low-Rank) structure: $A = \Lambda - PQ^T$ where Λ is diagonal and $P, Q \in \mathbb{R}^{N \times r}$ with $r \ll N$. The HiPPO-LegS matrix has exactly this structure with r = 1. This structure enables efficient computation of the resolvent $(sI - A)^{-1}$ using the Woodbury identity, which is the key operation in the frequency-domain computation of the SSM kernel.

3.4 Mamba: Selective State Space Models

S4 and its immediate successors (S4D, H3, Hyena) are linear time-invariant (LTI) systems: the matrices A, B, C do not depend on the input u. This is mathematically clean and enables the O(L log L) convolution computation, but it is also a fundamental limitation: the system cannot selectively focus on or ignore specific inputs based on their content. Mamba (Gu and Dao, 2023) addresses this by making B, C (and optionally Δ) functions of the input, creating a selective SSM.

Selectivity mechanism. In Mamba, the discretisation step Δ and the input projection B and output projection C are computed from the current input u_t via learned linear projections:

$$\Delta_t = \mathrm{softplus}(W_\Delta u_t + b_\Delta) \in \mathbb{R}^d$$

$$B_t = W_B u_t \in \mathbb{R}^N, \quad C_t = W_C u_t \in \mathbb{R}^N$$

This input-dependence means that the state update x_t = Ā(Δ_t) x_{t-1} + B̄(Δ_t, B_t) u_t can selectively remember or forget based on the content of u_t. Large Δ_t → ∞ makes Ā → 0, I (the state is reset or accumulated depending on B_t); small Δ_t → 0 makes Ā → I (the state is preserved unchanged). The model can thus learn to ignore irrelevant tokens (small Δ_t, reset), accumulate relevant ones (large Δ_t with appropriate B_t), and selectively retrieve with learned C_t.

Theorem 3.4 (Mamba Selectivity and the Limitation of Linear Attention)

A linear time-invariant SSM with fixed A, B, C cannot solve the Selective Copying task: given a sequence of inputs $u_1, \ldots, u_L$ where a subset are marked as 'select', output those selected inputs in order. This task requires the system to condition its state update on the content of the input (whether it is marked), which is by definition not possible for a fixed-coefficient linear system.

Mamba's input-dependent Δ_t, B_t, C_t resolve this: a large Δ_t for marked inputs causes them to be written into the state (via B_t), while unselected inputs with small Δ_t leave the state unchanged. Selective Copying, Induction Heads, and similar tasks that require content-dependent memory are solved by Mamba but not by S4/Hyena. This theoretical gap corresponds to the empirical finding that Mamba matches or surpasses transformers on discrete sequence modelling tasks (language, DNA, time series) while using $O(L)$ rather than $O(L^2)$ memory.

Parallel scan for training. Mamba's input-dependent recurrence cannot be computed as a convolution (the kernel changes at every step). Naive sequential computation costs O(L) steps, unparallelisable. The key algorithmic innovation is the parallel prefix scan (also called scan or prefix sum), which computes the recurrence x_t = a_t x_{t-1} + b_t for input-dependent a_t, b_t in O(log L) parallel steps using a balanced binary tree reduction:

$$Scan(a, b): O_t = \prod_{\tau=1}^{t} a_\tau \times x_0 + \Sigma_{\tau=1}^{t} (\prod_{s=\tau+1}^{t} a_s)\, b_\tau$$

The parallel scan is implemented as a custom CUDA kernel that performs the tree reduction in shared memory, achieving O(L) total work (optimal) in O(log L) parallel steps. This, combined with a hardware-efficient memory management strategy similar to Flash Attention (recomputing activations in the backward pass from the compressed recurrent state), gives Mamba a practical training cost of O(Ld) — similar to a transformer with $O(L^2d)$ attention replaced by O(Ld) selective scanning.

3.5 RWKV: Recurrence with Linear Attention

RWKV (Peng et al., 2023) takes a different approach to efficient sequence modelling: it reformulates linear attention as a recurrence, enabling the model to run as an RNN at inference while training in parallel as an attention model. The name RWKV stands for the four model components: Receptance (R), Weight (W), Key (K), Value (V).

The WKV operator. At time t, RWKV computes the attention-like output:

$$wkv_t = (\Sigma_{s<t} \exp(-(t-1-s)w + k_s)\, v_s + \exp(u + k_t)\, v_t) \,/\, (\Sigma_{s<t} \exp(-(t-1-s)w + k_s) + \exp(u + k_t))$$

where $w \in \mathbb{R}^d$ is a learned per-channel time decay vector (always negative so $\exp(-s \cdot w)$ decays with time), $u \in \mathbb{R}^d$ is a learned per-channel bonus for the current token, and k_t, v_t are the key and value at time t. This can be rewritten as a recurrence by defining:

$$a_t = \exp(-w) \odot a_{t-1} + \exp(k_t) \odot v_t, \quad b_t = \exp(-w) \odot b_{t-1} + \exp(k_t)$$

$$wkv_t = (a_{t-1} + \exp(u + k_t)\, v_t) \,/\, (b_{t-1} + \exp(u + k_t))$$

The recurrence is linear in (a_t, b_t), making it computable in O(1) per step at inference and parallelisable in O(L) via parallel prefix scan during training. RWKV-4 (the first production-scale version) achieved performance competitive with GPT-3 class transformers on language modelling benchmarks while using constant memory at inference, enabling deployment in memory-constrained environments.

Architecture	Training Cost	Inference Cost (per token)	Memory (inference)	Long-range Dependencies
Transformer	$O(L^2d)$	$O(Ld)$ with KV-Cache	$O(L)$ cache	Full $O(L)$ receptive field
LSTM	$O(Ld)$	$O(d^2)$ matrix multiply	$O(d)$	Exponential decay
S4	$O(L \log L \cdot d)$	$O(Nd)$	$O(N)$	Polynomial decay (HiPPO)
Mamba	$O(Ld)$ parallel scan	$O(Nd)$	$O(N)$	Selective: learned decay
RWKV	$O(Ld)$ prefix scan	$O(d)$	$O(d)$	Exponential decay (learnable w)
Hybrid (Jamba)	$O(L^2d) + O(Ld)$	$O(d) + O(Ld)$ hybrid	Reduced cache	Both: local + global

Chapter 4: Graph Neural Networks and the Weisfeiler-Leman Hierarchy

Graph-structured data is pervasive in science and technology: molecular graphs in drug discovery, social networks in recommendation systems, knowledge graphs in question answering, scene graphs in computer vision, and mesh graphs in physics simulation. Graph Neural Networks (GNNs) are the dominant framework for learning on such data. Unlike sequence or image data, graphs have no canonical ordering of nodes — any relabelling of nodes that preserves graph structure should produce equivalent outputs. This permutation symmetry is the fundamental inductive bias of GNNs, and the Weisfeiler-Leman (WL) graph isomorphism test provides the sharpest theoretical characterisation of how well GNNs can exploit it.

4.1 Message-Passing Neural Networks: The General Framework

A graph G = (V, E) consists of nodes V = {1, ..., n} with node features h_v^{(0)} ∈ ℝ^d and (optionally) edge features e_{uv} ∈ ℝ^{d_e} for edges (u, v) ∈ E. A Message-Passing Neural Network (MPNN, Gilmer et al., 2017) applies L rounds of neighbourhood aggregation:

$$m_v^{(t)} = AGG_t(\{MSG_t(h_v^{(t-1)}, h_u^{(t-1)}, e_{uv}) : u \in \mathcal{N}(v)\})$$

$$h_v^{(t)} = UPD_t(h_v^{(t-1)}, m_v^{(t)})$$

where AGG is a permutation-invariant aggregation function (sum, mean, max, attention-weighted sum), MSG is a message function, and UPD is an update function. After L rounds, a readout function produces the graph-level prediction: ŷ = READOUT({h_v^{(L)} : v ∈ V}). Specific choices of MSG, AGG, UPD recover all major GNN architectures: GCN (fixed normalised mean aggregation), GraphSAGE (learnable mean), GAT (attention-weighted), GIN (injective MLP aggregation), MPNN (explicit edge features).

Permutation equivariance. The MPNN framework is permutation equivariant by construction: reordering the nodes of G produces a reordering of the node representations {h_v^{(L)}}, and the graph-level readout (a permutation-invariant function of node representations) is permutation invariant. This is the correct inductive bias for graph-structured data: the graph's properties do not depend on how we label its nodes.

4.2 The Weisfeiler-Leman Test: Graph Isomorphism and Its Hierarchy

The Weisfeiler-Leman (WL) graph isomorphism test is a classical algorithm for determining whether two graphs are isomorphic. It provides the theoretical ceiling on the expressive power of message-passing GNNs. Understanding the WL test and its generalisations is the key to understanding both the fundamental limitations of standard GNNs and the design principles of more expressive architectures.

1-WL (Colour Refinement). The 1-WL algorithm assigns colours (labels) to nodes and iteratively refines them based on the multiset of neighbour colours:

$$c_v^{(t+1)} = HASH(c_v^{(t)}, \{\{c_u^{(t)} : u \in \mathcal{N}(v)\}\})$$

where HASH is an injective function on pairs (colour, multiset of colours). After stabilisation (at most n steps for n-node graphs), the histogram of final colours forms a graph signature. Two graphs are declared non-isomorphic if their signatures differ. The 1-WL test is: return 'possibly isomorphic' if the signatures match, 'non-isomorphic' if they differ.

Theorem 4.1 (GNN Expressiveness Equals 1-WL, Xu et al., 2019)

Any MPNN with a fixed number of layers L is at most as powerful as the 1-WL test in distinguishing non-isomorphic graphs. Moreover, there exists an MPNN (specifically, GIN — Graph Isomorphism Network) that is as powerful as 1-WL:

$h_v^{(t)} = MLP^{(t)}((1+\varepsilon^{(t)}) h_v^{(t-1)} + \Sigma_{u \in \mathcal{N}(v)} h_u^{(t-1)})$

where $\varepsilon^{(t)}$ is a learnable scalar. GIN is as powerful as 1-WL because its aggregation function is injective on multisets of node features: if the node features of two 1-hop neighbourhoods are different

as multisets, GIN produces different output features. Sum aggregation is injective on multisets of non-negative integer-valued features (and approximately injective for real-valued features); mean and max aggregation are not injective and strictly weaker than 1-WL.

Corollary: standard MPNNs (GCN, GraphSAGE, GAT with mean aggregation) are strictly less powerful than 1-WL. They cannot distinguish: (a) regular graphs of the same degree but different structure; (b) cycles and paths of the same length; (c) certain pairs of non-isomorphic graphs that 1-WL can distinguish.

Limitations of 1-WL. The 1-WL test itself fails to distinguish many pairs of non-isomorphic graphs. The classic example is two regular graphs with the same degree sequence but different structure (e.g., the 3-regular complete bipartite graph K_{3,3} and the 3-regular complete graph K_4, both with 6 nodes and 9 edges; or two 4-regular graphs on 6 nodes). More subtly, 1-WL cannot count cycles (it cannot distinguish graphs that differ only in their cycle structure), cannot count triangles (fundamental for social network analysis), and cannot compute graph properties such as the Weisfeiler-Leman dimension or tree-width.

4.3 The k-WL Hierarchy and Higher-Order GNNs

The k-dimensional Weisfeiler-Leman test (k-WL) operates on tuples of k nodes rather than individual nodes, providing a hierarchy of increasingly powerful graph isomorphism tests. The hierarchy is strict: for each k, there exist graphs that k-WL cannot distinguish but (k+1)-WL can.

k-WL definition. Assign initial colours $c^{(0)}((v_1,...,v_k)) = \text{HASH}(G[\{v_1,...,v_k\}])$ based on the isomorphism type of the induced subgraph on the k-tuple. Then iterate:

$$c^{(t+1)}(v_1,...,v_k) = HASH(c^{(t)}(v_1,...,v_k), \{\{c^{(t)}(v_1,...,v_{i-1},u,v_{i+1},...,v_k) : u \in V\}\} \text{ for } i=1,...,k)$$

Each step aggregates the colours of all k-tuples reachable by replacing one element of the current tuple with any node in the graph. The resulting colour histogram distinguishes more and more graph pairs as k increases.

Theorem 4.2 (WL Hierarchy is Strict)

For each $k \geq 1$, there exist non-isomorphic graphs G and H such that k-WL fails to distinguish them but (k+1)-WL succeeds. Moreover, k-WL is equivalent to the k-variable fragment of first-order logic with counting quantifiers (C_k) for graph properties. The entire hierarchy collapses in terms of decidability at $k = 2$ for vertex-transitive graphs (2-WL is not complete even for strongly regular graphs), and the hierarchy is not known to collapse at any finite k for general graphs.

The practical consequence for GNNs: standard MPNNs are at most 1-WL. Higher-order GNNs operating on k-tuples of nodes are equivalent to k-WL but have computational cost $O(n^k)$, making $k \geq 3$ prohibitively expensive for large graphs. The challenge is to design GNNs that exceed 1-WL expressiveness without the $O(n^k)$ cost.

4.4 Beyond 1-WL: Expressive GNN Architectures

Several architectures have been proposed to exceed the expressiveness of 1-WL without incurring the full $O(n^k)$ cost of higher-order GNNs. Each exploits a different structural feature of graphs that 1-WL cannot capture.

Random Node Initialisation (RNI). Assign each node a random feature from a continuous distribution before running MPNN. With probability 1, all nodes receive distinct features, making the MPNN effectively as powerful as an arbitrary isomorphism test (since distinct initial features break all symmetries). RNI achieves universal expressiveness in expectation at no architectural cost, but its randomness makes the output non-deterministic and requires averaging over multiple runs at inference, increasing cost by a factor equal to the number of samples.

Distance-WL and structural features. Augmenting node features with distance information — the shortest-path distance between each pair of nodes, or the distance from a set of landmark nodes — allows MPNNs to distinguish graphs based on their metric structure. Distance-WL (a GNN variant that uses

pairwise distances as edge features) can distinguish all pairs of graphs that 1-WL cannot distinguish using only local structure. The cost is $O(n^2)$ preprocessing for all-pairs shortest paths (Johnson's algorithm or BFS from each node).

NGNN, OSAN, and subgraph GNNs. A powerful family of approaches computes the representation of each node in the context of subgraphs: for each node v, consider the graph G_v obtained by marking v or deleting it, and run a base MPNN on G_v to obtain node-contextual representations. NGNN (Nested GNN) uses this idea with a hierarchy of subgraph MPNN calls. 3-WL-equivalent architectures can be obtained by considering all induced subgraphs on node pairs (DS-GNN, ESAN). The cost is O(n) base MPNN calls per graph, so $O(n^2)$ total for node-pair subgraphs.

Spectral GNNs and polynomial filters. An orthogonal approach uses the graph Laplacian $L = D - A$ (where D is the degree matrix) to define convolutional filters in the spectral domain. The graph Fourier transform $F = U^T$ (where $L = U\Lambda U^T$) diagonalises convolution, and a graph convolution is a polynomial filter $h(\Lambda) = \Sigma_k \theta_k \Lambda_k$ applied in the spectral domain: $y = U\, h(\Lambda)\, U^T x$. Polynomial filters of degree K are equivalent to K-hop message passing in the spatial domain (due to the Cayley-Hamilton theorem applied to the Laplacian), and spectral GNNs with learnable polynomial filters (ChebNet, GPRGNN, BernNet) are at most 1-WL equivalent. Filters using the full eigenvector basis (SIGN, filtering across all eigenvalues) are more expressive but require the $O(n^3)$ eigendecomposition.

Architecture	WL Power	Node Cost	Edge Cost	Key Mechanism
GCN, GraphSAGE (mean)	< 1-WL	$O(\|E\|)$	$O(\|E\|)$	Mean aggregation (non-injective)
GIN	= 1-WL	$O(\|E\|)$	$O(\|E\|)$	Sum aggregation (injective multisets)

GAT	≤ 1-WL	O(|E|)	O(|E|)	Attention-weighted mean (non-injective)
RNI + GIN	Universal (probabilistic)	O(|E|)	O(|E|)	Random node features
Subgraph GNNs (ESAN)	= 3-WL	O(n|E|)	O(n|E|)	Node-deletion subgraph ensemble
k-GNN (k ≥ 3)	= k-WL	O(n^k)	O(n^k)	k-tuple nodes
Spectral (full eigenbasis)	> 1-WL (partially)	$O(n^3 + n\|E\|)$	$O(n^3)$	Full Laplacian eigenvectors

4.5 Graph Transformers: Attention Meets Topology

The Transformer architecture naturally applies to graph-structured data if we treat the graph as a fully connected set of nodes and use the graph's edges as positional encodings or attention biases. Graph Transformers combine the permutation-equivariance of GNNs with the $O(n^2)$ full-graph attention of Transformers, producing architectures that exceed 1-WL expressiveness while remaining practically trainable for graphs with hundreds to thousands of nodes.

Graphormer. Ying et al. (2021) augment standard multi-head attention with three graph-structural features: (1) a centrality encoding based on degree (in-degree and out-degree for directed graphs) added to node features; (2) a spatial encoding that adds a learnable bias $A_{ij} = b_{\phi(i,j)}$ to the attention score between nodes i and j, where $\phi(i,j)$ is the shortest-path distance; (3) an edge encoding that averages learnable edge-type embeddings along the shortest path between i and j. Graphormer strictly surpasses 1-WL (because the distance-based bias encodes metric structure not captured by local neighbourhood aggregation) and achieves state-of-the-art results on molecular property prediction benchmarks.

GPS (General, Powerful, Scalable GNN). Rampášek et al. (2022) propose a principled decomposition of graph learning into two components: a local MPNN for aggregating neighbourhood information (providing strong structural inductive bias) and a global attention layer for long-range interactions (providing expressive power beyond 1-WL). The local and global components are interleaved in each layer, combined with positional encodings (Laplacian eigenvectors, random walk statistics) that encode spectral and topological structure. GPS achieves near-state-of-the-art across all standard graph learning benchmarks while remaining scalable to million-node graphs via sparse attention patterns.

Theorem 4.3 (Graph Transformers are Turing-Universal on Graphs)

A Graph Transformer with full ($O(n^2)$) attention, a sufficient number of layers, and positional encodings that distinguish all nodes (e.g., Laplacian eigenvectors in generic position) can compute any function of the graph that is computable by a Turing machine, up to approximation error. The proof is constructive: the attention mechanism can implement arbitrary set operations on node features; the positional encodings provide unique node identities; and the feed-forward sublayers provide arbitrary pointwise computation. This universality contrasts sharply with standard MPNNs, which are bounded by 1-WL.

The cost of this universality is $O(n^2)$ attention computation. For large graphs ($n > 10{,}000$), this is intractable, and the design of expressive sub-quadratic graph attention is an open research problem. Current approaches include linearised graph attention (replacing softmax attention with kernel methods, as in Section 2.5 of this volume), random sparse attention masks, and hierarchical graph pooling followed by full attention at coarser resolutions.

Conclusions: The Mathematical Logic of Architecture Choice

The four chapters of this volume have developed a coherent framework for understanding neural network architectures as mathematical objects with precise structural properties, rather than engineering constructions chosen by empirical trial. Looking across the territory covered, three master themes emerge.

The first theme is the group-theoretic formulation of inductive biases. Every architecture makes claims about the symmetries of the problem — translation equivariance (CNNs), permutation equivariance (GNNs), rotation equivariance (SE(3)-GNNs), relative position equivariance (RoPE). These claims are either correct (in which case building them in reduces sample complexity proportionally to the group size) or incorrect (in which case they introduce structural bias that hurts generalisation on data that does not respect the assumed symmetry). The design question is not 'what inductive bias is convenient?' but 'what symmetry does the data distribution have?'

The second theme is the expressiveness-efficiency tradeoff, most sharply crystallised in the WL hierarchy for graph neural networks. Increasing expressiveness (from 1-WL to k-WL) costs an exponential factor O(n^k) in computation. The practical resolution — hybrid architectures (GPS, Graphormer), structural positional encodings, subgraph ensembles — achieves super-WL expressiveness at subexponential cost, but always with an overhead relative to the 1-WL baseline. The same tradeoff appears in the sequence modelling context: full attention ($O(L^2)$) versus linear attention ($O(L)$), with selective SSMs (Mamba) and recurrent architectures (RWKV) occupying different points on the expressiveness-efficiency frontier.

The third theme is the memory-computation duality that appears throughout: KV-Cache trades memory for inference speed; Flash Attention trades recomputation for memory; Mamba's parallel scan trades memory-bandwidth for FLOPs; the WL hierarchy trades node-tuple memory for graph

discriminating power. Every efficiency innovation in this chapter is, at its mathematical core, a rearrangement of the memory-computation tradeoff to match the constraints of the available hardware. This is a deep point: the dominant architectures of any generation are determined not only by the data structure of the problems but by the memory hierarchy of the hardware on which they run.

Part III will shift from the structural (what functions can architectures represent?) to the statistical (how do they generalise from finite data?), developing the mathematical theory of statistical learning, including VC dimension, Rademacher complexity, the double descent phenomenon, the Neural Tangent Kernel, Bayesian inference, and uncertainty quantification.

PART III

STATISTICAL LEARNING AND BAYESIAN INFERENCE

PAC Learning and VC Dimension • Rademacher Complexity
Neural Tangent Kernel • Double Descent • Gaussian Processes
Variational Inference • Uncertainty Quantification

Eight-Part Monograph Series on the Mathematical Principles of Neural Networks
Independent of Companion Volumes — Complete in Itself

Silvano Salvador

© Mathematics of Deep Learning Series

PART III

STATISTICAL LEARNING AND BAYESIAN INFERENCE

A rigorous treatment of learning theory, probabilistic inference, and uncertainty quantification

Preface: The Probabilistic Foundations of Learning

This monograph constitutes the third part of a eight-part treatise on the mathematics of artificial intelligence. It is designed to be read independently of the other volumes, though readers familiar with Parts I and II will recognise recurring themes — particularly the geometry of information and the role of symmetry — that now manifest in probabilistic rather than algebraic clothing.

The central questions addressed here are deceptively simple to state and remarkably deep to answer. When does a learning algorithm trained on finite data reliably predict unseen outcomes? How does one quantify the uncertainty that inevitably accompanies such predictions? When exact inference is intractable — as it almost always is for interesting models — what principled approximation schemes exist, and what guarantees do they carry?

The answer to the first question resides in **statistical learning theory**, which supplies a rigorous vocabulary for generalisation: VC dimension, Rademacher complexity, algorithmic stability, and the tantalising double-descent phenomenon that challenges classical intuitions about model complexity. The answers to the second and third questions reside in the complementary traditions of **Markov chain Monte Carlo** and **variational inference**, two great engines of modern probabilistic computing whose mathematical foundations are surprisingly distinct — one rooted in ergodic theory and stochastic dynamics, the other in convex optimisation and information geometry.

A coherent treatment of all these topics requires the reader to hold simultaneously in mind ideas from measure theory, differential geometry, functional analysis, and stochastic calculus. We attempt to provide the conceptual scaffolding that makes this synthesis possible, keeping rigour always in view while never letting formalism obscure the underlying ideas.

Chapter 1: Statistical Learning Theory

Statistical learning theory asks: under what conditions, and at what sample complexity, can a learning algorithm trained on a finite dataset reliably generalise to new data drawn from the same distribution? The framework we develop — rooted in Valiant's PAC learning model and Vapnik's structural risk minimisation — provides answers whose form is universal, applying to any hypothesis class and any loss function satisfying mild regularity conditions.

1.1 The PAC Learning Framework

Let $\mathcal{X}$ be an input space, $\mathcal{Y}$ an output space, and $\mathcal{H}$ a hypothesis class of functions $h: \mathcal{X} \to \mathcal{Y}$. A concept $c: \mathcal{X} \to \mathcal{Y}$ is to be learned from a sample $S = \{(x_1, y_1), \dots, (x_m, y_m)\}$ drawn i.i.d. from an unknown distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$. The **true risk** of a hypothesis is $L(h) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(h(x), y)]$ and the **empirical risk** is $\hat{L}(h) = (1/m)\,\Sigma_i\, \ell(h(x_i), y_i)$.

A learning algorithm A **PAC-learns** $\mathcal{H}$ if for every $\varepsilon, \delta > 0$ and every distribution $\mathcal{D}$, given a sample of size $m \geq m(\varepsilon, \delta)$, with probability at least $1 - \delta$ over the draw of S, the output hypothesis satisfies $L(A(S)) \leq \min_{h\in\mathcal{H}} L(h) + \varepsilon$. The quantity $m(\varepsilon, \delta)$ is the **sample complexity** — the number of examples needed to achieve accuracy ε with confidence $1 - \delta$.

Theorem 1.1 — Finite Hypothesis Class Bound

Let $\mathcal{H}$ be finite, $\ell \in [0,1]$. For any $\delta > 0$, with probability $\geq 1-\delta$ over $S \sim \mathcal{D}^m$, for all $h \in \mathcal{H}$:

$$L(h) \leq \hat{L}(h) + \sqrt{\left(\ln|\mathcal{H}| + \ln(1/\delta)\right) / (2m)}$$

Proof: By a union bound over $|\mathcal{H}|$ Hoeffding inequalities, one per hypothesis. The $|\mathcal{H}|$-term measures the complexity penalty for searching through the class. The bound is tight for the ERM on uniformly bad instances.

For infinite hypothesis classes, the finite-class bound is inapplicable. The first key insight of VC theory is that the effective size of a hypothesis class for a binary classification problem is governed not by its cardinality but by its capacity to shatter finite sets of points — the VC dimension.

1.2 VC Dimension and the Fundamental Theorem of Learning

The **VC dimension** $d_{VC}(\mathcal{H})$ is the largest integer d such that there exists a set of d points that $\mathcal{H}$ can shatter — label in all 2^d possible ways. A hypothesis class with

$d_{VC}(\mathscr{H}) = \infty$ is not PAC-learnable (in the distribution-free sense). The Sauer-Shelah lemma controls the growth of the shatter function:

Lemma 1.2 — Sauer–Shelah

If d_VC($\mathscr{H}$) = d, then the growth function satisfies:

$$\Pi_{\mathscr{H}}(m) = \max_{S \subseteq \mathscr{X},\ |S|=m} |\{h|_S : h \in \mathscr{H}\}| \le \Sigma_{i=0}^{d} C(m,i) \le (em/d)^d$$

The key transition: for $m \le d$, $\Pi_{\mathscr{H}}(m) = 2^m$ (exponential); for $m > d$, $\Pi_{\mathscr{H}}(m)$ grows polynomially. This phase transition controls learnability.

Theorem 1.3 — Fundamental Theorem of PAC Learning (VC)

For a binary classification class $\mathscr{H}$ with $d = d_{VC}(\mathscr{H}) < \infty$, ERM PAC-learns $\mathscr{H}$ with sample complexity:

$$m(\varepsilon, \delta) = \Theta\left((d + \ln(1/\delta)) / \varepsilon^2 \right)$$

Upper bound proof: Symmetrisation (ghost sample), then union over $\Pi_{\mathscr{H}}(2m)$ dichotomies, then Hoeffding. Lower bound: Sauer-Shelah + a specific hard distribution on a shattered set.

The generalisation bound implied by the VC theorem predicts that test error decreases as model complexity increases until a sweet spot, then rises. This underpins the classical bias-variance tradeoff. But empirical observation of modern over-parameterised networks violates this prediction in a systematic way — a discrepancy resolved by the double-descent phenomenon.

1.3 Rademacher Complexity

Rademacher complexity provides a data-dependent generalisation bound, often tighter than VC bounds. Define the **empirical Rademacher complexity**:

$$\hat{R}_S(\in) = {}_i\ _\sigma[\ sup_\{h_i\}\ (1/m)\ \Sigma\ \ {}_i\ \sigma\text{-}h(x\text{-})]$$

where $\sigma_i \sim Uniform(\{\pm 1\})$ are independent Rademacher variables. Intuitively, $\hat{R}_S(\mathscr{H})$ measures how well the best hypothesis in $\mathscr{H}$ can correlate with random noise — a higher value means more capacity to fit arbitrary labels.

Theorem 1.4 — Rademacher Generalisation Bound

For any $\mathscr{H}$ of functions $\mathscr{H} \to [0,1]$ and any $\delta > 0$, with probability $\ge 1-\delta$ over $S \sim \mathscr{D}^m$, for all $h \in$

$\mathscr{H}$:

$$L(h) \le \hat{L}(h) + 2R_m(\mathscr{H}) + \sqrt{(\ln(1/\delta) / (2m))}$$

where $R_m(\mathscr{H}) = \mathbb{E}_S[\hat{R}_S(\mathscr{H})]$ is the expected Rademacher complexity. Unlike VC bounds, this captures the interaction between $\mathscr{H}$ and the data distribution $\mathscr{D}$.

For neural networks, Rademacher bounds can be expressed in terms of operator norms of weight matrices. For a depth-L network with weight matrices $W_1, \ldots, W_L$ and 1-Lipschitz activations:

$$R_m(\not{}net) \le (2^{3/2} B_x) / \sqrt{m} \cdot \Pi_{l=1}^{L} \|W_l\|_{op} \cdot \sqrt{(\Sigma_l \|W_l\|_F^2 / \|W_l\|_{op}^2)}$$

where $B_x = max\ \|x\|$. This bound degrades with depth and the product of spectral norms, explaining why spectral normalisation (constraining $\|W_l\|_{op} \le 1$) is an effective regularisation technique. However, modern large networks routinely violate the conditions under which these bounds are non-vacuous, which motivates the algorithmic stability and PAC-Bayes approaches.

1.4 Algorithmic Stability and PAC-Bayes

Bousquet and Elisseeff's **algorithmic stability** framework measures generalisation through the sensitivity of an algorithm's output to replacing a single training point. Algorithm A is β-uniformly stable if for all datasets S, S' differing in one example:

$$\sup_{(x,y)} |\ell(A(S), x, y) - \ell(A(S'), x, y)| \le \beta$$

Theorem 1.5 — Stability Generalisation Bound

If A is β-uniformly stable with $\beta \le 1/(2m)$, then for any $\delta > 0$, with prob. $\ge 1-\delta$:

$$L(A(S)) \le \hat{L}(A(S)) + 2\beta + (4m\beta + 1)\sqrt{(\ln(1/\delta) / (2m))}$$

SGD with step size η satisfies $\beta = O(\eta T/m)$ after T steps. This connects optimisation trajectory length to generalisation, explaining why early stopping helps.

PAC-Bayes bounds take a Bayesian perspective. Given a prior P over hypotheses and a posterior Q chosen after seeing data:

Theorem 1.6 — McAllester PAC-Bayes Bound

For any prior P, with probability ≥ 1−δ over S ∼ $\mathcal{D}^m$, for all posteriors Q:

$$\mathbb{E}_Q[L(h)] \le \mathbb{E}_Q[\hat{L}(h)] + \sqrt{\left(D_{KL}(Q\|P) + \ln(m/\delta)\right) / (2m-1)}$$

This bound is non-vacuous even for infinite hypothesis classes because the KL term controls the effective complexity. PAC-Bayes is the only framework giving tight (single-digit percentage) error bounds for deep networks.

1.5 Double Descent and Benign Overfitting

The classical bias-variance tradeoff predicts a U-shaped test error curve as a function of model complexity: bias decreases while variance increases, and there exists an optimal complexity at the interpolation point. The double-descent phenomenon, empirically documented across linear models, kernel machines, and neural networks, reveals a **second descent** of test error beyond the interpolation threshold, fundamentally challenging classical learning theory.

The interpolation threshold corresponds to the number of parameters required to exactly fit the training data (training error = 0). Classical theory predicts diverging test error at this point. Instead, empirically: for interpolating models beyond this threshold, test error decreases again, approaching or surpassing the classical optimum.

Theorem 1.7 — Benign Overfitting in Linear Models (Bartlett et al. 2020)

Consider the minimum-ℓ^2-norm interpolating estimator $\hat{\beta} = X^{\dagger}y$ for $X \in \mathcal{D}^{n \times p}$ with $p > n$.

Under Gaussian design with effective rank $r_{eff} = (\sum_i \lambda_i)^2 / \sum_i \lambda_i^2$ of the covariance Σ:

$$\|\hat{\beta} - \beta^*\|_{\Sigma}^2 = O(n / r_{eff} + \text{noise} \cdot p/n)$$

Generalisation is benign (risk → 0 as $n,p \to \infty$) if and only if $r_{eff}/n \to \infty$ AND $p/n \to \infty$.

The condition $r_{eff}/n \to \infty$ requires the spectrum of Σ to be spread — a "spiked" spectrum leads to malignant overfitting.

The mechanism behind benign overfitting is the **implicit regularisation** of gradient descent: with zero initialisation, gradient descent converges to the minimum-ℓ^2-norm interpolating solution in the underdetermined linear case. For neural networks, the analogous result holds in the **Neural Tangent Kernel** regime: gradient flow converges to the minimum-NTK-norm interpolating function. The geometry of the NTK — its

effective rank and eigenspectrum — determines whether overfitting is benign or malignant.

1.6 The Neural Tangent Kernel

At infinite width, a randomly initialised neural network $f_\theta: \mathbb{R}^{n_0} \to \mathbb{R}$ converges to a Gaussian process, and during training with gradient flow the parameters barely move from initialisation — the **lazy training** or **NTK regime**. The Neural Tangent Kernel (Jacot et al. 2018) is:

$$\Theta(x, x') = \sum_p (\partial f_\theta(x)/\partial\theta_p)(\partial f_\theta(x')/\partial\theta_p)$$

Theorem 1.8 — NTK Convergence

For a fully-connected network of width n with Lipschitz activations σ, as n → ∞:

(i) The NTK Θ(θ, x, x') converges in probability to a deterministic kernel Θ∞(x, x')

(ii) Under gradient flow, Θ(θ(t), ·, ·) = Θ∞(·, ·) + O(1/√n) for all t ≥ 0

(iii) The network function f_θ(t) evolves as: df/dt = −Θ∞ (f − y), giving f(t) = (I−e^{−Θ∞t}) (Θ∞)⁻¹y

Convergence rate: min eigenvalue of Θ∞ on training set determines exponential convergence rate of training loss.

The NTK regime is mathematically clean but practically limited. In finite-width networks, features are learned — the kernel changes during training — and this **feature learning** is responsible for the empirical superiority of trained networks over kernel machines with fixed NTK. The catapult phase (initial rapid kernel change before lazy settling) and the edge-of-stability phenomenon (loss oscillating near the sharpness threshold $2/\eta$ with η the learning rate) are active research frontiers beyond the NTK theory.

Chapter 2: Optimisation and Loss Landscape

The loss landscape of a neural network is a high-dimensional function whose global geometry determines which optimisation algorithms succeed. The landscape of modern over-parameterised networks is profoundly non-convex, yet in practice gradient-based methods reliably find solutions with near-zero training loss that also generalise. Understanding why requires a synthesis of dynamical systems theory, random matrix theory, and topological analysis.

2.1 Stochastic Gradient Descent as a Dynamical System

Let $L(\theta) = \mathbb{E}_{z\sim\mathcal{D}}[\ell(f_\theta, z)]$ be the population loss. Gradient descent iterates $\theta_{t+1} = \theta_t - \eta\nabla L(\theta_t)$ but computing ∇L exactly is infeasible. SGD uses a mini-batch estimate:

$$\theta_{t+1} = \theta_t - \eta\,\hat{g}_t, \quad \text{where } \hat{g}_t = \frac{1}{|B_t|}\sum_{z\in B_t}\nabla\ell(f_{\theta_t}, z)$$

The noise $\hat{g}_t - \nabla L(\theta_t)$ has zero mean (unbiased) and covariance approximately $C(\theta_t)/|B|$ where $C(\theta) = \mathbb{E}[\nabla\ell\,\nabla\ell^T] - \nabla L\,\nabla L^T$ is the gradient covariance. In continuous time, SGD is approximated by the Langevin SDE:

$$d\theta = -\nabla L(\theta)\,dt + \sqrt{\eta/|B|}\,C(\theta)^{1/2}\,dW_t$$

This connection to Langevin dynamics reveals that SGD implicitly samples from the Gibbs distribution $\pi(\theta) \propto \exp(-L(\theta)/T)$ with temperature $T = \eta/(2|B|)$. Larger learning rates or smaller batches correspond to higher temperature — more exploration. The flat-minimum preference of SGD (its tendency to converge to wide valleys in the loss landscape rather than sharp minima) arises from this temperature: sharp minima occupy small measure under the Gibbs distribution at any positive temperature.

Theorem 2.1 — SGD Flat-Minimum Bias (Zhu et al. 2019)

Let θ^* be a local minimum. The expected escape time from a δ-neighbourhood scales as:

$$\tau_{\text{escape}} \propto \exp(\Delta L / T) = \exp(|B|\,\Delta L / \eta)$$

where ΔL is the loss barrier height at the boundary. Flatter minima (same depth but wider basin) have lower barriers in most directions, hence shorter escape time — SGD spends more time in flat basins proportional to their volume × depth tradeoff.

2.2 Adaptive Methods: Adam, AdaGrad, and RMSProp

The pathologies of SGD — sensitivity to learning rate, slow convergence in directions of small curvature — motivated adaptive methods that rescale each gradient coordinate by a running estimate of its magnitude. The unifying framework:

$$\theta_{t+1} = \theta_t - \eta \times m_t \oslash (v_t^{1/2} + \varepsilon)$$

where m_t is the first moment estimate (gradient), v_t is the second moment estimate (squared gradient), and $\oslash$ is element-wise division. **AdaGrad** sets $v_t = \Sigma_{s\le t}\ \hat{g}_s^2$ (cumulative sum — decreasing effective LR, never recovers); **RMSProp** uses an exponential moving average $v_t = \beta v_{t-1} + (1-\beta)\hat{g}_t^2$; **Adam** combines both:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\hat{g}_t, \quad v_t = \beta_2 v_{t-1} + (1-\beta_2)\hat{g}_t^2$$

$$\hat{m}_t = m_t/(1-\beta_1^t), \quad \hat{v}_t = v_t/(1-\beta_2^t) \quad \textit{(bias correction)}$$

AdamW separates weight decay from the adaptive scaling: instead of adding $\lambda\theta$ to the gradient before adaptation (which shrinks the effective weight decay for high-variance parameters), it applies $\theta_t \leftarrow (1 - \lambda\eta)\theta_{t-1}$ as a multiplicative step. This decoupling is crucial for correct regularisation of embedding layers and attention weights, which typically have much smaller gradient variance than MLP weights.

Theorem 2.2 — Adam Convergence (Non-convex)

For L smooth ($\|\nabla^2 L\| \le \beta$) with gradients g_t having bounded variance σ^2:

$$(1/T)\ \Sigma_{t=1}^{T}\ \mathbb{E}[\|\nabla L(\theta_t)\|^2] \le O(\sqrt{(d \ln T)} / \sqrt{T})$$

where d is the parameter dimension. The ln T factor (logarithmic regret) arises from the time-varying denominator $\hat{v}_t^{1/2}$.

Caveat: This bound is for fixed β_1, β_2 and does not account for the warm-up phase. Empirically, the first few thousand steps are critical and not well-modelled by asymptotic theory.

2.3 Loss Landscape Geometry: Saddle Points, Flat Minima, and the Mode Connectivity

In high-dimensional non-convex optimisation, the dominant obstruction to convergence is not local minima but **saddle points** — critical points where the Hessian has both positive and negative eigenvalues. In a p-dimensional landscape, a random critical point is a local minimum with probability exponentially small in p. Almost all critical points with loss $L > L^*$ are saddle points.

Theorem 2.3 — Saddle Point Avoidance (Lee et al. 2016)

For gradient flow on a Morse function (all critical points non-degenerate), the set of initial conditions converging to a strict saddle point has Lebesgue measure zero. More strongly, for smooth gradient descent with appropriate step size, all stable limit points of gradient descent are local minima.

Practical implication: SGD noise (or any perturbation) guarantees escape from saddle points, justifying the empirical observation that SGD finds minima rather than getting stuck at saddles.

Mode connectivity (Draxler et al. 2018, Garipov et al. 2018) reveals that independently trained solutions in the loss landscape are connected by paths of approximately constant loss. Specifically, for overparameterised networks there exist piecewise-linear paths $\gamma: [0,1] \to \mathbb{R}^p$ between any two converged solutions θ_A and θ_B such that $L(\gamma(t)) \approx \max(L(\theta_A), L(\theta_B))$ for all t. This mode connectivity implies that the loss landscape does not fragment into isolated good solutions but rather contains a large connected manifold of good minima — the **solution manifold hypothesis**.

The sharpness-awareness of a minimum is measured by the maximum eigenvalue of the Hessian $\lambda_{max}(\nabla^2 L(\theta))$. SAM (Sharpness-Aware Minimisation) solves:

$$\min_\theta \max_{\|\varepsilon\| \le \rho} L(\theta+\varepsilon) \quad \text{approximately:} \quad \min_\theta L(\theta) + \rho \|\nabla L(\theta)\|$$

The perturbation $\hat{\varepsilon} = \rho \nabla L / \|\nabla L\|$ maximising local loss has a clean geometric interpretation: it pushes θ along the steepest ascent direction on the loss surface, forcing the optimiser to find flatlands rather than ridges. SAM's efficacy is partly explained by a PAC-Bayes bound: the expected loss under a spherical Gaussian perturbation of radius ρ is a valid generalisation bound, and SAM directly minimises this quantity.

2.4 The Edge of Stability

An unexpected phenomenon in full-batch gradient descent — the **edge of stability** (Cohen et al. 2021) — challenges the standard convergence theory. For learning rate η, gradient descent is classically guaranteed to decrease loss when $\lambda_{max}(\nabla^2 L) < 2/\eta$. Yet experiments show that during training, λ_{max} rises precisely to $2/\eta$ and hovers there, while the loss still decreases. The mechanism:

(1) **Progressive sharpening**: gradient descent naturally moves towards sharper regions of the landscape; (2) At the edge $\lambda_{max} \approx 2/\eta$, the dynamics enter a regime of **loss oscillations** with period approximately 2 (chaotic but loss-decreasing on

average); (3) The bulk structure of the Hessian (not just λ_max) controls the effective step taken in each direction, allowing stable descent in low-curvature subspaces even when the sharpest direction is unstable.

Chapter 3: Kernel Methods — RKHS, SVM, and Maximum Mean Discrepancy

Kernel methods provide a principled framework for non-linear learning through a central algebraic device: representing the model as a function in a Reproducing Kernel Hilbert Space defined implicitly by a positive-definite kernel function. This framework — now understood as a classical limit of infinitely wide neural networks — achieves optimal generalisation bounds with explicit geometric interpretation.

3.1 Reproducing Kernel Hilbert Spaces

A **reproducing kernel Hilbert space** (RKHS) $\mathscr{H}_k$ associated with a positive-definite kernel $k: \mathscr{X} \times \mathscr{X} \to \mathbb{R}$ is a Hilbert space of functions $f: \mathscr{X} \to \mathbb{R}$ satisfying the reproducing property:

$$f(x) = \langle f, k(\cdot, x) \rangle_{\mathscr{H}_k} \quad \text{for all } f \in \mathscr{H}_k \text{ and } x \in \mathscr{X}$$

This property has profound consequences. First, **Mercer's theorem** provides a spectral representation: for a continuous positive-definite kernel on a compact domain, there exist eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots > 0$ and eigenfunctions ψ_j such that $k(x, x') = \sum_j \lambda_j \psi_j(x) \psi_j(x')$. The RKHS is then the image of ℓ^2 under the map $(c_j) \mapsto \sum_j c_j \sqrt{\lambda_j}\, \psi_j$. Second, the **kernel trick** allows computing inner products in $\mathscr{H}_k$ without explicit feature maps:

$$\langle \varphi(x), \varphi(x') \rangle_{\mathscr{H}} = k(x, x') \quad \text{where} \quad \varphi(x) = k(\cdot, x) \in \mathscr{H}_k$$

Theorem 3.1 — Representer Theorem (Schölkopf–Smola–Müller)

For any loss $L(f(x_1), \ldots, f(x_m), y_1, \ldots, y_m)$ and RKHS regulariser $\Omega(\|f\|_{\mathscr{H}_k})$:

$$f^* = \operatorname{argmin}_{f \in \mathscr{H}_k} L(f(x_1), \ldots, f(x_m), y) + \Omega(\|f\|_{\mathscr{H}_k})$$

has the finite expansion: $f^*(\cdot) = \sum_{i} \alpha_i\, k(\cdot, x_i)$

Proof: Any component of f orthogonal to $\operatorname{span}\{k(\cdot, x_i)\}$ neither changes the loss (which depends only on evaluations at training points) nor decreases $\|f\|$, so the optimum has zero orthogonal component.

Common kernels and their RKHS interpretation: the **Gaussian RBF** $k(x,x') = \exp(-\|x-x'\|^2/2\sigma^2)$ corresponds to an infinite-dimensional feature space (entire function class of analytic functions); the **polynomial kernel** $k(x,x') = (x^T x' + c)^d$ corresponds to

all monomials of degree up to d; the **arc-cosine kernel** $k_n(x,x')$ of Cho and Saul is exactly the NTK of an infinite-width network with specific activations.

3.2 Support Vector Machines and Structural Risk Minimisation

The Support Vector Machine arises from Vapnik's **structural risk minimisation** principle: instead of minimising empirical risk alone (ERM), jointly minimise empirical risk and a measure of model complexity. For binary classification with labels $y \in \{\pm 1\}$, the hard-margin SVM seeks:

$$\min_{w,\,b} \ \tfrac{1}{2}\|w\|^2 \quad \text{subject to} \quad y_i(w \cdot x_i + b) \geq 1 \ \ \forall i$$

The geometric interpretation: the constraint ensures that all training points lie outside a margin band of width $2/\|w\|$ around the decision boundary. Minimising $\tfrac{1}{2}\|w\|^2$ therefore **maximises the margin**. The dual problem:

$$\max_{\alpha} \ \sum_i \alpha_i - \tfrac{1}{2} \sum_{ij} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \quad \text{s.t.} \ \ \alpha_i \geq 0, \ \ \sum_i \alpha_i y_i = 0$$

The KKT conditions show that only points on the margin boundary (support vectors) have $\alpha_i > 0$ — the model complexity depends only on the number of support vectors, not the total training set size. The soft-margin extension (C-SVM) introduces slack variables $\xi_i \geq 0$ and penalty $C \sum_i \xi_i$, trading margin width for tolerance of misclassification.

Theorem 3.2 — SVM Generalisation Bound via Margin

For a linear classifier with margin ϱ, with probability $\geq 1-\delta$:

$$L(h) \leq \hat{L}_\gamma(h) + O\left(\hat{R}(\|w\|/\gamma) + \sqrt{(\ln(1/\delta)/m)} \right)$$

where $\hat{L}_\gamma$ is the γ-margin training error and $\hat{R}$ is the Rademacher complexity bounded by the margin. This shows SVMs directly minimise the generalisation-relevant quantity, not just empirical risk.

3.3 Maximum Mean Discrepancy

Maximum Mean Discrepancy (MMD) is a kernel-based measure of distributional distance with applications from two-sample testing to generative modelling evaluation. Given probability distributions P and Q on $\mathcal{X}$, and a unit ball in RKHS $\mathcal{H}_k$:

$$\mathrm{MMD}^2(P, Q) = \sup_{f \in \mathcal{H}_k,\ \|f\| \leq 1} \left(\mathbb{E}_{x \sim P}[f(x)] - \mathbb{E}_{y \sim Q}[f(y)] \right)^2$$

The variational definition yields a computable estimator: $MMD^2(P,Q) = \mathbb{E}_{x,x' \sim P}[k(x,x')] - 2\mathbb{E}_{x \sim P, y \sim Q}[k(x,y)] + \mathbb{E}_{y,y' \sim Q}[k(y,y')]$. This is a U-statistic with well-understood asymptotic distribution under $H_0: P = Q$ (the MMD statistic follows a weighted sum of χ^2_1 distributions under the null), enabling rigorous hypothesis testing.

Theorem 3.3 — MMD Metrises Weak Convergence

For a characteristic kernel k (e.g., Gaussian RBF), MMD(P,Q) = 0 $\iff$ P = Q.

Furthermore, MMD(P_n, P) $\to$ 0 iff $P_n \Rightarrow P$ (weak convergence) when $\mathcal{X}$ is compact.

This distinguishes MMD from moment-matching: characteristic kernels capture ALL moments of the distribution simultaneously, not just finite-order ones.

In machine learning, MMD appears as: (1) the training objective for Maximum Mean Discrepancy Networks (a precursor to GANs); (2) the divergence in InfoVAE, replacing the KL term in the ELBO; (3) a diagnostic for covariate shift and domain adaptation; (4) the Stein discrepancy (a related kernel measure) appears in SVGD (Stein Variational Gradient Descent), a particle-based approximate inference method.

3.4 Kernel Stein Discrepancy and SVGD

The **Kernelised Stein Discrepancy** (KSD) measures how far a distribution q is from a target p without requiring samples from p — only access to $\nabla \log p$. For Stein operator $\nabla_p f(x) = \nabla_x \log p(x)\, f(x)$ + $\nabla_x f(x)$:

$$KSD^2(q, p) = \mathbb{E}_{x,x' \sim q}[u_p(x, x')]$$

where the kernel Stein function $u_p(x, x')$ depends only on $\nabla \log p$ and k. This enables testing whether $q = p$ using only samples from q — powerful for evaluating MCMC convergence without knowing p's normalisation constant.

Stein Variational Gradient Descent (SVGD, Liu and Wang 2016) uses functional gradient descent on the KL divergence: starting from particles $x_i \sim q_0$, the update $x_i \leftarrow x_i + \varepsilon\, \varphi^*(x_i)$ with optimal velocity field $\varphi^*(x) = \mathbb{E}_{y \sim q}[k(y,x) \nabla_y \log p(y) + \nabla_y k(y,x)]$ yields the steepest descent in W_2 distance on the manifold of smooth densities. SVGD bridges particle methods and variational inference: with one particle it becomes MAP estimation; with many particles it converges to the posterior.

Chapter 4: Markov Chain Monte Carlo

Monte Carlo methods address the fundamental challenge of Bayesian computation: given a posterior distribution $\pi(\theta) \propto p(\theta) \times p(\text{data}|\theta)$ that is analytically intractable, how do we compute expectations $\mathbb{E}_\pi[f(\theta)]$? Markov chain Monte Carlo constructs an ergodic Markov chain with stationary distribution π, exploiting the ergodic theorem to convert time averages into space averages. The various MCMC algorithms differ in how they construct this chain, with fundamental tradeoffs between mixing time, computational cost per step, and applicability.

4.1 Metropolis-Hastings: The Master Algorithm

The Metropolis-Hastings (MH) algorithm constructs a reversible Markov chain on any measurable space by accepting or rejecting proposed moves with a carefully designed acceptance probability. Given proposal kernel $q(\theta'|\theta)$, from current state θ_t:

$\theta' \sim q(\cdot|\theta_t)$ (propose)

$\alpha = \min(1, \pi(\theta')q(\theta_t|\theta')/\pi(\theta)q(\theta'|\theta_t))$ (acceptance ratio)

$\theta_{t+1} = \theta'$ with prob α, else θ_t (accept/reject)

Theorem 4.1 — MH Correctness

The MH chain satisfies detailed balance: $\pi(\theta)T(\theta'|\theta) = \pi(\theta')T(\theta|\theta')$ for all θ, θ'.

Proof: For $\alpha < 1$ states, $\pi(\theta)q(\theta'|\theta)\alpha(\theta,\theta') = \pi(\theta')q(\theta|\theta') \times \pi(\theta)q(\theta'|\theta)/\pi(\theta')q(\theta|\theta') = \pi(\theta)q(\theta'|\theta)$, and $\alpha(\theta',\theta) = 1$ by symmetry of the ratio, giving detailed balance. The chain is π-irreducible if $\text{supp}(q(\cdot|\theta)) = \text{supp}(\pi)$ for all θ.

The critical design choices are the proposal q and the scale at which it operates. For a Gaussian random walk proposal with covariance $\Sigma\Delta$, the optimal acceptance rate is approximately 23.4% in high dimensions (Roberts-Gelman-Gilks 1997), achieved with $\Sigma\Delta = (2.38^2/d)\,\Sigma_\pi$ where Σ_π is the posterior covariance. This scaling law reveals a fundamental tension: proposals that are too narrow mix slowly (high acceptance rate but small moves); proposals that are too wide rarely accept. The RWMH mixing time scales as $O(d)$ for isotropic posteriors — polynomial in dimension, but prohibitive for thousands of parameters.

4.2 Hamiltonian Monte Carlo

Hamiltonian Monte Carlo (HMC) escapes the random-walk scaling by proposing moves along Hamiltonian trajectories. Augment the parameter space with auxiliary momentum $p \sim \mathcal{N}(0, M)$ and define the Hamiltonian:

$$H(\theta, p) = -\log \pi(\theta) + \tfrac{1}{2} p^T M^{-1} p \quad \text{(potential + kinetic energy)}$$

Hamiltonian dynamics $d\theta/dt = \partial H/\partial p = M^{-1}p$, $dp/dt = -\partial H/\partial \theta = \nabla \log \pi(\theta)$ preserve both H (energy conservation) and volume (Liouville's theorem). Discretising with the leapfrog integrator:

$$p_{t+\varepsilon/2} = p_t + (\varepsilon/2)\nabla \log \pi(\theta_t)$$

$$\theta_{t+\varepsilon} = \theta_t + \varepsilon M^{-1} p_{t+\varepsilon/2}$$

$$p_{t+\varepsilon} = p_{t+\varepsilon/2} + (\varepsilon/2)\nabla \log \pi(\theta_{t+\varepsilon})$$

After L leapfrog steps, the proposed (θ', p') is accepted via an MH step correcting for discretisation error. For well-tuned step size ε, acceptance rates are close to 1 (typically 65-80%), and the mixing time scales as $O(d^{1/4})$ — a dramatic improvement over RWMH's $O(d)$.

Theorem 4.2 — HMC Mixing (Mangoubi-Smith 2019)

For log-concave target π with condition number κ (ratio of max to min curvature), HMC with leapfrog integration mixes in:

$$O(\kappa^{1/2} d^{1/4}) \text{ steps (vs } O(\kappa d) \text{ for Langevin, } O(\kappa d) \text{ for RWMH)}$$

The condition number κ measures posterior anisotropy; adaptive preconditioning (using estimated $M = \Sigma_\pi$) reduces the effective κ to 1.

4.3 NUTS: No-U-Turn Sampler

The No-U-Turn Sampler (Hoffman-Gelman 2014) automates the critical tuning parameters of HMC — step size ε and trajectory length L. The number of leapfrog steps L is chosen adaptively by detecting when the trajectory begins to "turn around" (the U-turn criterion):

$$\text{stop when } (\theta_L - \theta_0) \cdot p_L < 0 \text{ or } (\theta_L - \theta_0) \cdot p_0 < 0$$

NUTS builds a binary tree of proposals by doubling the trajectory in the direction that does not violate the U-turn criterion, then samples uniformly from the resulting set (the "slice" in the joint (θ, p) space). The resulting algorithm satisfies detailed balance and is provably at least as efficient as optimal fixed-length HMC. Dual averaging

(Nesterov 2009) adapts the step size ε during a warmup phase to achieve a target acceptance probability (typically 0.8).

NUTS is the default sampler in Stan and PyMC, enabling Bayesian inference on complex hierarchical models with thousands of parameters. Its main limitation is the gradient computation cost per step — for large neural networks with millions of parameters, NUTS is computationally prohibitive, motivating the SGLD approach.

4.4 Stochastic Gradient Langevin Dynamics

SGLD (Welling and Teh 2011) brings Bayesian deep learning into computational reach by replacing the exact gradient with a mini-batch estimate and eliminating the MH accept/reject step. Starting from the overdamped Langevin SDE:

$$d\theta = \tfrac{1}{2}\nabla \log \pi(\theta)\, dt + dW_t \implies \theta_{t+1} = \theta_t + (\varepsilon_t/2)\nabla \log \pi(\theta_t) + \sqrt{\varepsilon_t}\, \xi_t$$

where $\xi_t \sim \mathcal{N}(0, I)$. Replacing $\nabla \log \pi$ with a stochastic estimate on a mini-batch B_t:

$$\nabla \log \pi(\theta) = \nabla \log p(\theta) + (N/|B|) \sum_{i \in B} \nabla \log p(x_i \mid \theta)$$

Theorem 4.3 — SGLD Convergence

Under log-concavity and Lipschitz gradient assumptions, SGLD with step size $\varepsilon \propto t^{-1/3}$ satisfies:

$$W_2(\rho_t, \pi) \le O(t^{-1/3} + \text{noise_bias})$$

where noise_bias comes from mini-batch gradient noise. The step size must be annealed to zero for exact convergence; a fixed step size yields a biased stationary distribution within $O(\varepsilon)$ of π in W_2.

In practice: use fixed ε as an approximate Bayesian sampler and diagnose convergence via Stein discrepancy or kernel two-sample tests.

4.5 Parallel Tempering and Replica Exchange

Highly multimodal posteriors — common in neural network parameter spaces, mixture models, and protein folding — defeat all unimodal MCMC algorithms by creating bottlenecks at low-probability barriers between modes. **Parallel tempering** (replica exchange MCMC) simultaneously runs chains at different temperatures $T_1 < T_2 < \ldots < T_K$ targeting the tempered distributions $\pi_k(\theta) \propto \pi(\theta)^{1/T_k}$.

At high temperature, barriers are smoothed and modes mix freely. Periodically, adjacent chains attempt swap moves with acceptance probability:

$$\alpha_{swap} = \min(1, [\pi_{k+1}(\theta_k)/\pi_k(\theta_k)] \times [\pi_k(\theta_{k+1})/\pi_{k+1}(\theta_{k+1})])$$

$$= \min(1, \exp((1/T_{k+1} - 1/T_k)(\log \pi(\theta_k) - \log \pi(\theta_{k+1}))))$$

Accepted swaps propagate low-temperature solutions that were found at high temperature, enabling the cold chain to jump between modes that are separated by barriers too high for direct Metropolis moves. The optimal temperature ladder (for spherical Gaussians) is geometric: $T_k = T_1^{(K-k)/(K-1)}$, which ensures roughly constant swap rates of approximately 23.4%. For real-world posteriors, adaptive temperature scheduling based on measured swap rates is standard.

Theorem 4.4 — Parallel Tempering Mixing Enhancement

For a bimodal target with mode separation Δ and barrier height B, RWMH mixing time is O(exp(B/T)).

Parallel tempering with K temperatures mixing to the cold chain in $O(K \times \text{mixing_at_}T_1)$ steps, where $T_1 \geq B$ means the hottest chain mixes in O(poly(d)) steps.

Optimal K = O(log(B/T_cold) × d^{1/2}) temperature levels achieve polynomial mixing.

Chapter 5: Variational Inference

Variational inference transforms posterior computation from a sampling problem into an optimisation problem. By restricting the posterior approximation to a tractable family and maximising a lower bound on the log evidence, variational methods scale to large datasets and complex models where MCMC is computationally prohibitive. The price paid is approximation bias from the restricted family; the gain is computational speed of several orders of magnitude.

5.1 The Evidence Lower Bound (ELBO)

For a model with observations x, latent variables z, and parameters θ, the log-evidence $\log p_\theta(x)$ is intractable because it requires integrating over z. The ELBO decomposes this quantity:

$$\log p_\theta(x) = \mathbb{E}_{q_\varphi(z|x)}[\log p_\theta(x,z) - \log q_\varphi(z|x)] + D_{KL}(q_\varphi(z|x) \,\|\, p_\theta(z|x))$$

$$\geq \mathcal{L}(\theta, \varphi; x) \equiv \mathbb{E}_{q_\varphi}[\log p_\theta(x|z)] - D_{KL}(q_\varphi(z|x) \,\|\, p(z))$$

The ELBO decomposes as a reconstruction term (expected log-likelihood) minus a KL regulariser. The bound is tight when $q_\varphi(z|x) = p_\theta(z|x)$ — the exact posterior — which is the condition that VI is trying to approximate. Maximising over θ and φ simultaneously performs both model learning and posterior approximation.

The KL divergence $D_{KL}(q_\varphi \,\|\, p)$ in the ELBO uses the **forward KL** (also called exclusive or mean-seeking). Optimising this form encourages q_φ to be zero wherever p is zero, and to cover all modes of p — resulting in broad, diffuse approximations that may miss fine-grained posterior structure. The alternative **reverse KL** $D_{KL}(p \,\|\, q_\varphi)$ (inclusive, mass-covering) penalises q_φ for being small where p is large, producing mode-seeking behaviour. Most VI literature uses forward KL for computational tractability.

5.2 Coordinate Ascent Variational Inference (CAVI)

Under the **mean-field assumption**, the variational posterior factorises over J disjoint groups: $q(z) = \prod_{j=1}^{J} q_j(z_j)$. CAVI optimises each factor in turn while holding others fixed:

Theorem 5.1 — Optimal Mean-Field Factor

Under the mean-field assumption, the ELBO-optimal jth factor satisfies:

q*_j(z_j) ∝ exp(𝔼_{q_{−j}}[log p(z, x)])

where the expectation is over all variables z_k (k ≠ j) under their current variational distributions.

CAVI: iterate this update for j = 1, …, J until convergence. Each step monotonically increases the ELBO (coordinate ascent in a jointly non-concave objective).

CAVI was the dominant VI algorithm for latent Dirichlet allocation (LDA), mixture models, and hierarchical Bayesian models throughout the 2000s. Its strength is the clean, closed-form updates available when the model is in the **conjugate exponential family**: if $p(z_j \mid z_{-j}, x)$ belongs to the exponential family, the optimal q_j^* has the same form and the natural parameters update as linear functions of expectations under other factors.

5.3 Black-Box Variational Inference (BBVI)

CAVI requires closed-form coordinate updates unavailable for non-conjugate models. **Black-box VI** (Ranganath et al. 2014) estimates the ELBO gradient via Monte Carlo sampling. The gradient with respect to φ:

$$\nabla_\varphi \mathscr{L} = \mathbb{E}_{q_\varphi}[\nabla_\varphi \log q_\varphi(z) \times (\log p(x, z) - \log q_\varphi(z))]$$

(the **score function estimator** or REINFORCE gradient). This estimator is unbiased but has extremely high variance: the term $\log p(x,z) - \log q_\varphi(z)$ (the "control variate") must be near zero for low variance, but there is no natural baseline without model-specific information.

The **reparametrisation trick** provides a low-variance alternative when q_φ admits a differentiable reparametrisation $z = g(\varepsilon; \varphi)$ with $\varepsilon \sim \mathcal{N}(0, I)$:

$$\nabla_\varphi \mathscr{L} = \mathbb{E}_{\varepsilon \sim \mathcal{N}(0,I)}[\nabla_\varphi \log p(x, g(\varepsilon;\varphi)) - \nabla_\varphi \log q_\varphi(g(\varepsilon;\varphi))]$$

This gradient flows through the sampler via the Jacobian $\partial g / \partial \varphi$, giving much lower variance. It requires that q_φ be reparametrisable — satisfied by Gaussians (location-scale family), Gamma, Beta, and many other distributions, but not discrete distributions. The reparametrisation trick is the cornerstone of the VAE and of modern stochastic variational inference.

Theorem 5.2 — Reparametrisation Gradient Variance

For the reparametrisation estimator ĝ_rep and score function estimator ĝ_score:

Var[ĝ_rep] = O(‖J‖²_F × Var_ε[log p gradient])

Var[ĝ_score] = Var_z[(log p − log q) × score]

The reparametrisation estimator is typically orders of magnitude lower variance because it exploits the smoothness of p with respect to φ, while the score estimator treats the model as a black box.

5.4 Importance-Weighted Autoencoder (IWAE)

The ELBO uses a single-sample Monte Carlo estimate of $\log p(x)$. The **Importance-Weighted Autoencoder** (Burda et al. 2015) tightens this bound by averaging over K importance-weighted samples:

$$\mathcal{L}_K(\theta, \varphi; x) = \mathbb{E}_{\{z_1,\ldots,z_K \sim q_\varphi(\cdot|x)\}}[\log (1/K) \Sigma_k w_k]$$

$$\text{where } w_k = p_\theta(x, z_k) / q_\varphi(z_k|x) \quad \text{(unnormalised importance weights)}$$

Theorem 5.3 — IWAE Bound Hierarchy

The bounds satisfy: $\log p_\theta(x) \geq \mathcal{L}_{K+1} \geq \mathcal{L}_K \geq \mathcal{L}_1 = \text{ELBO}$, with equality as $K \to \infty$.

Proof: By Jensen's inequality applied to the concave log function:

$\mathcal{L}_K = \mathbb{E}[\log (1/K)\Sigma w_k] \leq \log \mathbb{E}[(1/K)\Sigma w_k] = \log p_\theta(x)$

The gap $\mathcal{L}_\infty - \mathcal{L}_K = O(\text{Var}[w]/K)$ by the delta method, measuring importance weight degeneracy.

The IWAE gradient has a crucial pathology: for large K, the gradient signal is dominated by the single sample with the highest weight, causing φ to update almost exclusively based on high-probability regions of $p_\theta(z|x)$. This **posterior collapse** problem means that tighter bounds do not necessarily lead to better representation learning. The IWAE-DReG (Doubly-Reparametrised Gradient) estimator corrects this by stopping gradients through the weight normalisation:

$$\nabla_\varphi \mathcal{L}_K^{DReG} = \mathbb{E}[\Sigma_k \tilde{w}_k^2 \nabla_\varphi \log q_\varphi(z_k|x)] \quad \text{where } \tilde{w}_k = w_k / \Sigma_j w_j \text{ (stopped)}$$

5.5 Normalising Flows for Variational Inference

Mean-field variational inference is limited by the expressiveness of the factored Gaussian posterior. **Normalising flows** enrich the variational family by composing a

sequence of invertible transformations $z_K = f_K \circ \ldots \circ f_1(z_0)$ starting from a simple distribution $z_0 \sim q_0$:

$$\log q_K(z_K) = \log q_0(z_0) - \Sigma_t \log |\det J_t|$$

The ELBO with flow-based posterior:

$$\mathscr{L} = \mathbb{E}_{q_0}[\log p(x, z_K) - \log q_0(z_0) + \Sigma_t \log |\det J_t|]$$

Common flow architectures for VI include **IAF** (Inverse Autoregressive Flows), which achieve efficient sampling but slow density evaluation; **Planar Flows** with transformations $f(z) = z + uh(w^T z + b)$ (one-rank perturbations); and **Radial Flows** expanding/contracting around reference points. The expressiveness-computation tradeoff determines which flows are practical for VI: sampling (for the ELBO) should be fast, while log-density evaluation (for gradient computation) must also be tractable.

Chapter 6: Uncertainty Quantification

A well-calibrated predictive system should know what it does not know. Uncertainty quantification (UQ) methods transform a point-prediction neural network into a system that assigns reliable confidence levels to its outputs — confidence that tracks true predictive uncertainty rather than being an artefact of the training objective or architecture. The methods range from principled Bayesian approaches to practical ensemble and conformal methods.

6.1 Bayesian Neural Networks

A **Bayesian neural network** (BNN) places a prior $p(\omega)$ over network weights ω and performs inference via the posterior $p(\omega \mid \mathcal{D}) \propto p(\omega)p(\mathcal{D} \mid \omega)$. Predictions are marginalised over the posterior:

$$p(y^* \mid x^*, \mathcal{D}) = \int p(y^* \mid x^*, \omega)\, p(\omega \mid \mathcal{D})\, d\omega$$

This integral is intractable for neural networks. The key challenges are: (1) the posterior $p(\omega \mid \mathcal{D})$ is high-dimensional, multimodal, and highly non-Gaussian; (2) standard MCMC (NUTS) does not scale to millions of parameters; (3) mean-field VI underestimates posterior uncertainty because the factored Gaussian approximation is too simple. SGLD addresses (2) at the cost of exact samples; structured VI (including matrix-Gaussian posteriors) partially addresses (3).

The practical utility of BNNs lies in the decomposition of predictive uncertainty into **aleatoric uncertainty** (irreducible noise in the data: $\mathbb{E}_\omega[\mathrm{Var}_{y \mid x,\omega}[y]]$) and **epistemic uncertainty** (model uncertainty reducible with more data: $\mathrm{Var}_\omega[\mathbb{E}_{y \mid x,\omega}[y]]$). For out-of-distribution (OOD) inputs, epistemic uncertainty should be high — a well-calibrated BNN warns when it is being asked to extrapolate beyond training data.

6.2 Laplace Approximation

The Laplace approximation fits a Gaussian to the posterior at the MAP estimate. Given $\hat{\omega} = \arg\max \log p(\omega \mid \mathcal{D})$:

$$p(\omega \mid \mathcal{D}) \approx \mathcal{N}(\hat{\omega}, H^{-1}) \quad \text{where} \quad H = -\nabla^2_{\omega} \log p(\omega \mid \mathcal{D})\big|_{\omega=\hat{\omega}}$$

The Hessian H at the optimum is positive definite (MAP is a maximum), giving a valid Gaussian approximation. Predictive distribution:

$$p(y^* \mid x^*, \mathcal{D}) \approx \int p(y^* \mid x^*, \omega)\, \mathcal{N}(\omega;\, \hat{\omega}, H^{-1})\, d\omega$$

For neural networks with millions of parameters, computing and inverting the full Hessian is infeasible. Kronecker-Factored Laplace (**KFAC-Laplace**) approximates the Hessian block-diagonally per layer as a Kronecker product $H^{(l)} \approx A^{(l)} \otimes S^{(l)}$ — the same factorisation as K-FAC. Diagonal Laplace (**DiagLaplace**) uses only the diagonal of H, equivalent to independent Gaussian uncertainty per weight.

Theorem 6.1 — Laplace Approximation Quality

The Laplace approximation is exact at the optimum in the limit of infinite data (posterior concentrates at MAP). The first-order error term is:

$$D_{KL}(p_{Laplace} \| p_{posterior}) = O(\|\nabla^3 \log p\| / n^{1/2})$$

The approximation fails when: (i) the MAP is not the global mode, (ii) the posterior is multimodal, (iii) the curvature at the MAP does not represent the global geometry.

For OOD detection: the Laplace predictive variance is large when the test point lies in a region where the linearised network features are dissimilar from training data.

6.3 Deep Ensembles

Deep Ensembles (Lakshminarayanan et al. 2017) trains M independently initialised and trained networks and combines their predictions:

$$\bar{p}(y^* \mid x^*) = (1/M) \Sigma_m \, p_{\theta_m}(y^* \mid x^*)$$

For a Gaussian output head, the ensemble mean and variance are:

$$\bar{\mu} = (1/M) \Sigma \, \mu_m(x^*), \quad \bar{\sigma}^2 = (1/M) \Sigma(\sigma_m^2(x^*) + \mu_m(x^*)^2) - \bar{\mu}^2$$

The second term is the **epistemic uncertainty** from disagreement between ensemble members. Despite having no Bayesian justification, deep ensembles are empirically the strongest known UQ method — consistently outperforming BNNs, Laplace approximations, and MC Dropout across benchmarks. The likely explanation: the SGD loss landscape has multiple distinct good minima corresponding to qualitatively different function representations; independent training initialises near different basins, and the ensemble captures genuine diversity.

Theorem 6.2 — Ensemble Variance as Epistemic Uncertainty

For M independent models $\theta_1, \ldots, \theta_M \sim p(\theta \mid \circledcirc)$, the ensemble variance equals the posterior predictive variance:

$$\mathrm{Var}[\bar{y}] = \mathbb{E}[\mathrm{Var}[y \mid x, \theta]] + \mathrm{Var}[\mathbb{E}[y \mid x, \theta]] \quad \text{(bias-variance decomposition)}$$

The first term is aleatoric; the second is epistemic. With independent training, the M models approximate independent posterior samples — empirically confirmed by comparing ensemble calibration to approximate Bayes.

6.4 Conformal Prediction

Conformal Prediction (Vovk, Gammerman, Shafer 1999) provides distribution-free, finite-sample coverage guarantees for predictive sets without any assumptions on the data distribution beyond exchangeability. For a user-specified confidence level $1-\alpha$, conformal prediction produces a set $C(x_{n+1})$ that contains the true label with probability at least $1-\alpha$:

$$P(y_{n+1} \in C(x_{n+1})) \geq 1 - \alpha$$

Theorem 6.3 — Conformal Coverage Guarantee

Let $\{(x_1,y_1),\ldots,(x_n,y_n), (x_{n+1},y_{n+1})\}$ be exchangeable. For any conformity score $s(x,y)$:

Let $\hat{q} = (\lceil (n+1)(1-\alpha) \rceil / n)$-quantile of $\{s(x_i,y_i)\}_{i=1}^{n}$

Then $C(x_{n+1}) = \{y : s(x_{n+1}, y) \leq \hat{q}\}$ satisfies $P(y_{n+1} \in C(x)) \geq 1-\alpha$.

The guarantee is exact: $P(y_{n+1} \in C) = \lceil (n+1)(1-\alpha) \rceil / (n+1)$ — never more conservative than necessary.

The conformity score $s(x, y)$ is a non-conformity measure: for regression, $s(x,y) = |y - f(x)|$; for classification with softmax output, $s(x,y) = 1 - p_f(y|x)$. **Split conformal** divides data into calibration and test sets — the simplest and most practical variant. **Full conformal** retrains for each test point (exact but expensive). **Mondrian conformal** provides class-conditional coverage by computing quantiles separately per class.

Conformal risk control (Angelopoulos et al. 2022) generalises the coverage guarantee to arbitrary monotone risk functionals: for any user-specified risk R (error rate, false negative rate, expected set size), one can calibrate the prediction set to satisfy $\mathbb{E}[R(C(x_{n+1}))] \leq \alpha$. This extends conformal coverage to structured prediction (segmentation masks, graph outputs) and complex risk measures.

6.5 Calibration Assessment and the Reliability Diagram

A classifier is **perfectly calibrated** if among all instances where it predicts confidence p, exactly fraction p are correct:

$$P(y = \hat{y} \mid \hat{p}(\hat{y} \mid x) = p) = p \quad \text{for all } p \in [0,1]$$

The **Expected Calibration Error** (ECE) estimates this empirically by binning predictions into M equal-width intervals and computing the weighted average gap:

$$ECE = \Sigma_m \, |B_m|/n \times |acc(B_m) - conf(B_m)|$$

where $acc(B_m) = correct_in_bin \, / \, |B_m|$ and $conf(B_m) = mean(\hat{p})$ in bin m. Standard deep networks trained with cross-entropy loss are systematically overconfident: predicted confidence is higher than empirical accuracy. **Temperature scaling** (dividing logits by a scalar $T > 1$ before softmax) is the simplest post-hoc calibration: one parameter adjusted on a held-out calibration set, consistently outperforming more complex methods. The theoretical reason: T effectively broadens the predictive distribution, trading accuracy for calibration.

UQ Method	Coverage Guarantee	Scalability	Key Assumption	OOD Detection
Full BNN (NUTS)	Exact Bayes	Very low (d < 1000)	Prior specification	Excellent
SGLD / SGHMC	Approximate Bayes	Medium (millions)	Ergodicity + step size	Good
Laplace (KFAC)	Quadratic approx	High	Unimodal posterior	Moderate
MC Dropout	Approx. VI	High	Dropout ↔️ GP prior	Moderate
Deep Ensembles	None (empirical)	Medium (M models)	Independent minima	Excellent
Conformal	Distribution-free	Very high	Exchangeability only	Via coverage gap
Temperature Scaling	None (post-hoc)	Highest	Single scalar bias	None directly

MATHEMATICS OF DEEP LEARNING

PART IV

THE GENERATIVE PARADIGM

Variational Autoencoders · GANs · Normalising Flows · Score-Based Models
Diffusion · Flow Matching · Latent Diffusion · EBMs · Self-Supervised Learning

Eight-Part Monograph Series on the Mathematical Principles of Neural Networks
Independent of Companion Volumes — Complete in Itself

Silvano Salvador

© Mathematics of Deep Learning Series

PART IV

THE GENERATIVE PARADIGM

Variational Autoencoders · GANs · Normalising Flows · Score-Based Models
Diffusion · Flow Matching · Latent Diffusion · EBMs · Self-Supervised Learning

© Mathematics of Deep Learning Series

Silvano Salvador

Preface: The Problem of Generation — Why Predecessors Failed

Historical and Motivational Context. *Before 2013, the dominant approaches to generative modelling were Gaussian Mixture Models, Restricted Boltzmann Machines (RBMs), and explicitly-specified graphical models. Each failed in a characteristic way that reveals precisely what was missing. GMMs require the number of modes to be specified in advance and cannot represent multimodal distributions in high dimensions without exponentially many components. RBMs, developed by Smolensky (1986) and popularised by Hinton and Salakhutdinov (2006), can represent complex distributions but training via Contrastive Divergence is notoriously unstable: the MCMC chain used to estimate the negative phase gradient mixes slowly in high-dimensional spaces, producing biased gradient estimates whose bias grows as the model improves. Deep Boltzmann Machines addressed depth but compounded the sampling problem. Pixel RNNs and PixelCNNs (van den Oord et al., 2016) achieved exact likelihoods but generated samples autoregressively—one pixel at a time—making generation quadratically slow in image resolution. The central insight of the VAE, GAN, and flow families is that* learning to generate *and* learning to evaluate density *are separate problems, and separating them unlocks fundamentally different algorithmic strategies.*

To generate is, in a precise mathematical sense, to sample from a probability distribution over a structured, high-dimensional space. The challenge is that such distributions—over natural images, molecular structures, natural language utterances—are supported on low-dimensional manifolds embedded in spaces of dimension $D = 10^6$ or more, and are accessible only through finite samples without any analytic description. This part of the treatise examines the mathematical architectures that have transformed this challenge into practical reality.

The progression from Variational Autoencoders to Generative Adversarial Networks to Normalising Flows to Score-Based Diffusion Models and Flow Matching is not merely chronological. It represents a sequence of fundamentally different mathematical answers—each generation revealing a new geometric or probabilistic structure, each resolving pathologies of its predecessor at the cost of introducing new ones.

The unifying mathematical thread is *optimal transport*: the problem of finding the most efficient coupling between a simple reference distribution and the complex target. Dif-

fusion models learn this transport via a stochastic denoising path; flow matching via a deterministic vector field; GANs adversarially; VAEs via variational lower bounds. The Wasserstein distance, the Benamou–Brenier dynamic formulation, and the Monge–Kantorovich problem are the backbone unifying these diverse approaches.

Cross-Reference to Other Parts. The information geometry developed in Part I—the Fisher metric, the exponential manifold, and the notion of the natural gradient—provides the geometric language for understanding why KL divergence appears naturally in the VAE objective and why the choice of divergence profoundly affects training dynamics. The manifold hypothesis discussed in Part II, which posits that high-dimensional data concentrates near low-dimensional submanifolds, is the fundamental motivation for latent variable models. The PAC-Bayes bounds of Part III make precise the sense in which the ELBO is a compression objective.

Variational Autoencoders

Historical and Motivational Context. *The intellectual lineage of the VAE begins with Helmholtz (1866), who proposed that perception is a form of probabilistic inference: the brain maintains internal models (the "generative model") and perception is the process of inverting them (the "recognition model"). This idea was formalised computationally by Hinton and Zemel (1994) in the Helmholtz Machine, which learned separate recognition and generative networks. The key obstacle was the intractability of the posterior $p(z|x)$: the recognition network approximated it, but training required alternating between "wake" (learning the generative model) and "sleep" (learning the recognition model from generative samples) phases that were not end-to-end differentiable. The VAE (Kingma and Welling, 2013; Rezende, Wierstra and Wierlings, 2014) resolved this by two simultaneous insights: the reparametrisation trick, which makes sampling differentiable, and the ELBO, which converts the intractable log-likelihood into a tractable lower bound optimised simultaneously over both networks. The paper was rejected from ICLR 2013 on first submission due to concerns about approximation quality; its resubmission became one of the most cited papers in machine learning.*

The Variational Autoencoder (Kingma and Welling, 2013) is simultaneously a deep latent variable model, an amortised variational inference algorithm, and a representation learning system. Its mathematical elegance lies in the joint treatment of generation and inference within a single end-to-end differentiable framework governed by the Evidence Lower Bound.

The Latent Variable Model and the ELBO: Full Derivation

Let $z \in \mathbb{R}^d$ be a latent variable with prior $p(z) = \mathcal{N}(0, I_d)$ and let $x \in \mathbb{R}^D$ be observed, modelled by a decoder network $p_\theta(x|z)$. The joint is $p_\theta(x, z) = p_\theta(x|z)p(z)$ and the marginal $p_\theta(x) = \int p_\theta(x|z)p(z)\,dz$ is intractable.

We derive the ELBO from first principles. For any distribution $q_\phi(z|x)$, decompose the

log-marginal by inserting $q_\phi(z|x)/q_\phi(z|x)$ inside the integral:

$$\begin{aligned}\log p_\theta(x) &= \log \int p_\theta(x|z)p(z)\,dz = \log \int \frac{p_\theta(x|z)p(z)}{q_\phi(z|x)} q_\phi(z|x)\,dz \\ &= \log \mathbb{E}_{q_\phi(z|x)}\left[\frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}\right] \geq \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}\right] \end{aligned} \tag{1}$$

where the last step is Jensen's inequality applied to the concave function log. Separating the logarithm:

$$\begin{aligned}\mathcal{L}(\theta,\phi;x) &= \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z) + \log p(z) - \log q_\phi(z|x)] \\ &= \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{q_\phi(z|x)}{p(z)}\right] \\ &= \underbrace{\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]}_{\text{reconstruction}} - \underbrace{D_{\mathrm{KL}}(q_\phi(z|x)\|p(z))}_{\text{regularisation}} \end{aligned} \tag{2}$$

The gap between $\log p_\theta(x)$ and $\mathcal{L}(\theta,\phi;x)$ is exactly $D_{\mathrm{KL}}(q_\phi(z|x)\|p_\theta(z|x)) \geq 0$, confirming that $\mathcal{L}$ is a lower bound that becomes tight when the approximate posterior equals the true posterior.

For a diagonal Gaussian encoder $q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \mathrm{diag}(\sigma^2_\phi(x)))$ and standard normal prior, the KL term has the closed form:

$$D_{\mathrm{KL}}(\mathcal{N}(\mu,\sigma^2 I)\|\mathcal{N}(0,I)) = \frac{1}{2}\sum_{j=1}^{d}\left(\mu_j^2 + \sigma_j^2 - \log\sigma_j^2 - 1\right) \tag{3}$$

Derivation of the closed-form KL. Using the general formula $D_{\mathrm{KL}}(\mathcal{N}(\mu_1,\Sigma_1)\|\mathcal{N}(\mu_2,\Sigma_2)) = \frac{1}{2}\big[\mathrm{tr}(\Sigma_2^{-1}\Sigma_1) + (\mu_2-\mu_1)^T\Sigma_2^{-1}(\mu_2-\mu_1) - d + \log(\det\Sigma_2/\det\Sigma_1)\big]$, with $\mu_1 = \mu$, $\Sigma_1 = \mathrm{diag}(\sigma^2)$, $\mu_2 = 0$, $\Sigma_2 = I$: $\mathrm{tr}(\Sigma_1) = \sum_j \sigma_j^2$, the quadratic term is $\|\mu\|^2$, and $\log\det\Sigma_2/\det\Sigma_1 = -\sum_j \log\sigma_j^2$. Summing gives the formula above.

The Reparametrisation Trick

Optimising $\mathcal{L}$ with respect to ϕ requires backpropagating through the sampling operation $z \sim q_\phi(z|x)$, which is non-differentiable. The reparametrisation trick resolves this by writing $z = \mu_\phi(x) + \sigma_\phi(x)\odot\varepsilon$ with $\varepsilon \sim \mathcal{N}(0,I)$, relocating the randomness to a fixed distribution:

$$\nabla_\phi \mathbb{E}_{q_\phi}[f(z)] = \mathbb{E}_\varepsilon\big[\nabla_\phi f\big(\mu_\phi(x) + \sigma_\phi(x)\odot\varepsilon\big)\big] \tag{4}$$

This transformation produces unbiased gradient estimates. The variance of the reparametrised estimator is polynomial in $\|\nabla_z f\|$, compared to the score function (REINFORCE) estimator whose variance is quadratic in the magnitude of f itself, often orders of magnitude larger.

Theorem 1.1 (Rate–Distortion Interpretation of the ELBO)**.** *The ELBO decomposes as: $\mathcal{L} = -\mathbb{E}[\textit{distortion}] - \textit{rate}$, where $\textit{distortion} = -\log p_\theta(x|z)$ is the reconstruction error and*

rate $= D_{\mathrm{KL}}(q_\phi(z|x)\|p(z))$ *measures the number of bits used by the latent code. The* β*-VAE generalises this to* $\mathcal{L}_\beta = -\mathbb{E}[\textit{distortion}] - \beta \cdot \textit{rate}$*.* $\beta > 1$ *encourages disentangled representations; the trade-off is fundamental—high decoder expressiveness drives the rate to zero (posterior collapse), while high* β *compresses representations at the cost of reconstruction quality.*

Posterior Collapse and Hierarchical VAEs

Posterior collapse occurs when the decoder $p_\theta(x|z)$ becomes powerful enough to model x without reference to z. The posterior $q_\phi(z|x)$ collapses to the prior and the KL term vanishes, producing a local ELBO maximum with zero mutual information between z and x. This is not a training failure in the standard sense—the ELBO is being maximised—but a degenerate solution where the latent code carries no information.

Hierarchical VAEs address posterior collapse by stacking multiple latent layers $z_L \to \cdots \to z_1 \to x$. The ELBO telescopes as $\mathcal{L} = \mathbb{E}[\log p(x|z_1)] - \sum_\ell D_{\mathrm{KL}}(q_\phi(z_\ell|z_{\ell-1}, x)\|p_\theta(z_\ell|z_{\ell-1}))$. NVAE, VDVAE, and Diffusion VAE achieve state-of-the-art density estimation by combining hierarchical structure with residual encoder–decoder networks. The aggregated posterior $q_\phi(z) = \frac{1}{N}\sum_i q_\phi(z|x_i)$ must match the prior for clean generation; the *holes problem* arises when regions of $p(z)$ not covered by any $q_\phi(z|x_i)$ produce poor samples.

> **Cross-Reference to Other Parts.** The posterior collapse phenomenon is directly related to the over-parameterisation analysis in Part III: when the decoder has capacity far exceeding the data complexity, it lies in the jamming-above phase (see Part VI, Chapter 1) and can trivially satisfy the reconstruction objective without using the latent code. The information-bottleneck principle of Part III formalises the rate–distortion trade-off in terms of the mutual information $I(Z; X)$.

Generative Adversarial Networks

> **Historical and Motivational Context.** *The GAN (Goodfellow et al., 2014) emerged from a thought experiment posed at a bar in Montréal: could two networks compete in such a way that the competition itself drives both to improve, without requiring any explicit density estimation? The idea was met with scepticism—Yann LeCun called it "the most interesting idea in machine learning in the last ten years"—but early training was famously unstable, producing blurry outputs and mode-collapsed samples. The first two years of GAN research consisted almost entirely of empirical tricks: feature matching, minibatch discrimination, one-sided label smoothing, and DCGAN architectural guidelines. The theoretical breakthrough came with WGAN (2017), which replaced JS divergence with the Wasserstein distance and provided the first rigorous explanation of why the original GAN objective fails and what to do about it. WGAN transformed GAN training from an art into a science.*

The GAN framework (Goodfellow et al., 2014) transforms generation into a two-player zero-sum game, eliminating the need for explicit density estimation or variational bounds.

The Minimax Game and JS Divergence

Given data distribution p_r and generator distribution $p_g = G_{\#}p_z$ (the pushforward of noise under G), the GAN objective is:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_r}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))] \tag{5}$$

Theorem 2.1 (GAN Optimal Discriminator and Global Minimum)**.** *For fixed G, the optimal discriminator is $D^*_G(x) = p_r(x)/(p_r(x) + p_g(x))$. Substituting D^*_G yields the reduced objective $C(G) = -\log 4 + 2 \cdot \mathrm{JSD}(p_r \| p_g)$, where* JSD *is the Jensen–Shannon divergence. The global minimum $C(G^*) = -\log 4$ is attained iff $p_g = p_r$. However, when p_r and p_g have disjoint supports—typical when p_g is supported on a low-dimensional manifold—*JSD $= \log 2$ *is constant and gradients vanish, creating the* vanishing gradient pathology *of the original GAN objective.*

Proof of the vanishing gradient result. When $\mathrm{supp}(p_r) \cap \mathrm{supp}(p_g) = \emptyset$, the optimal discriminator achieves perfect classification: $D^*(x) = 1$ for $x \in \mathrm{supp}(p_r)$ and $D^*(x) = 0$ for $x \in \mathrm{supp}(p_g)$. The JSD between two distributions with disjoint supports equals $\log 2$ (maximum possible), independent of the distance between them. Therefore $\nabla_G C(G) = 0$, and the generator receives no gradient signal. Since high-dimensional manifolds generically intersect in a set of measure zero, this is the default condition early in training, explaining the empirical instability of the original GAN.

Wasserstein GAN and the Earth Mover Distance

WGAN (Arjovsky et al., 2017) replaces JS divergence with the Wasserstein-1 distance:

$$W(p_r, p_g) = \inf_{\gamma \in \Pi(p_r, p_g)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|] \tag{6}$$

By the Kantorovich–Rubinstein duality: $W(p_r, p_g) = \sup_{\|f\|_L \le 1} \mathbb{E}_{x\sim p_r}[f(x)] - \mathbb{E}_{y\sim p_g}[f(y)]$ over 1-Lipschitz functions. WGAN-GP enforces the Lipschitz constraint via a gradient penalty: $\mathcal{L}_{\mathrm{GP}} = \mathbb{E}_{\hat{x}}[(\|\nabla_{\hat{x}} f_w(\hat{x})\|_2 - 1)^2]$ where $\hat{x}$ is sampled along interpolations between real and generated points.

Theorem 2.2 (WGAN Gradient Properties)**.** *The Wasserstein distance satisfies: (i) $W(p_r, p_g)$ is continuous everywhere and differentiable almost everywhere, even when supports are disjoint. (ii) $\nabla_\theta W(p_r, p_{g_\theta})$ is always informative—the gradient does not vanish regardless of the discriminator quality. (iii) W correlates monotonically with sample quality empirically, unlike JSD which saturates to* $\log 2$ *once supports separate.*

Cross-Reference to Other Parts. The Wasserstein distance is the key object in optimal transport theory, which Part I introduces via the Monge–Kantorovich problem. The flow matching approach of Chapter 7 exploits the Benamou–Brenier dynamic formulation of W_2, making the connection between transport theory and generative modelling fully explicit. The Rényi divergences discussed in Part I, which include the KL and JS divergences as special cases, give a unified view of why different divergence choices lead to different training dynamics.

StyleGAN, Progressive Growing, and Latent Disentanglement

StyleGAN (Karras et al., 2019) introduces an intermediate latent space $\mathcal{W}$ obtained from noise z via a learned mapping network $f : \mathcal{Z} \to \mathcal{W}$. Styles $w \in \mathcal{W}$ are injected at each resolution level via Adaptive Instance Normalisation (AdaIN):

$$\text{AdaIN}(x_i, w) = y_{s,i} \frac{x_i - \mu(x_i)}{\sigma(x_i)} + y_{b,i}, \quad (y_s, y_b) = A(w) \tag{7}$$

The $\mathcal{W}$ space achieves lower perceptual path length (PPL) than $\mathcal{Z}$, indicating more disentangled interpolations where each direction corresponds to an independent semantic attribute. StyleGAN2 eliminates blob artefacts via revised normalisation; StyleGAN3 replaces the feature map with continuous signal representations, achieving true equivariance to translation and rotation.

GAN Training Dynamics, Mode Collapse, and Spectral Normalisation

Mode collapse arises because the generator gradient $\nabla_\theta \mathbb{E}_z[-\log D(G_\theta(z))]$ pushes G toward regions where D assigns high probability; if D concentrates mass on a few modes, G satisfies this constraint by covering only those modes: $\text{supp}(p_g) \subsetneq \text{supp}(p_r)$.

Spectral normalisation (Miyato et al., 2018) enforces the 1-Lipschitz constraint by normalising each weight matrix by its spectral norm: $\tilde{W} = W / \sigma_{\max}(W)$, where $\sigma_{\max}(W)$ is estimated by the power iteration method. For a network with L layers, $\|f\|_L \leq \prod_\ell \sigma_{\max}(W_\ell)$, so normalising each layer to unit spectral norm bounds the global Lipschitz constant by 1.

Normalising Flows

Historical and Motivational Context. *The idea of learning a distribution by finding a diffeomorphism to a simple reference distribution dates at least to Tabak and Vanden-Eijnden (2010). Early flows (NICE, 2014; RealNVP, 2017) used coupling layers with deliberately restricted Jacobian structure; Glow (2018) scaled these to high-resolution images. The key theoretical limitation—that a single continuous flow cannot disconnect the topology of the base distribution—was understood early: a flow from a connected $\mathcal{N}(0, I)$ cannot produce a distribution supported on disconnected components without infinite depth. Continuous Normalising Flows (Chen et al., 2018) overcame depth limitations by framing the flow as an ODE, but training required expensive numerical ODE solving. This computational cost motivated flow matching (Lipman et al., 2022), which trains CNFs by directly regressing on the vector field rather than through the ODE.*

Normalising flows construct an explicit bijection between a simple base distribution and the data distribution, enabling exact likelihood computation and exact sampling.

The Change-of-Variables Formula

Let $z \sim p_z(z)$ be a base distribution and $f : \mathbb{R}^d \to \mathbb{R}^d$ a smooth bijection. The density of the pushforward $x = f(z)$ is:

$$\log p_x(x) = \log p_z(f^{-1}(x)) + \log \left| \det J_{f^{-1}}(x) \right| \tag{8}$$

where $J_{f^{-1}}$ is the Jacobian of f^{-1}. For a composition of flows $f = f_K \circ \cdots \circ f_1$, the log-likelihood decomposes as $\log p_x(x) = \log p_z(z_0) + \sum_k \log |\det J_{f_k^{-1}}(z_k)|$.

Theorem 3.1 (Triangular Jacobian and Efficient Determinant)**.** *If $f : \mathbb{R}^d \to \mathbb{R}^d$ has a triangular Jacobian matrix J_f with diagonal entries $(\partial f_i/\partial z_i)$, then $\det(J_f) = \prod_i (\partial f_i/\partial z_i)$, computable in $O(d)$. Coupling layers exploit this by partitioning $z = (z_{1:d/2}, z_{d/2+1:d})$ and defining: $x_{1:d/2} = z_{1:d/2}$, $x_{d/2+1:d} = z_{d/2+1:d} \odot \exp(s(z_{1:d/2})) + t(z_{1:d/2})$. The Jacobian is block-triangular giving $\log |\det J| = \sum_i s_i(z_{1:d/2})$.*

Coupling Layers: RealNVP and Glow

RealNVP (Dinh et al., 2017) implements affine coupling layers with alternating masks to ensure all dimensions are transformed. Glow (Kingma and Dhariwal, 2018) extends RealNVP with three components:

- **ActNorm**: affine normalisation with per-channel scale and bias initialised from the first minibatch, so the initial activations are standardised. This replaces batch normalisation in flows, where batch-dependent normalisation would break the invertibility guarantee.
- **1×1 invertible convolutions**: generalising channel permutations; the log-determinant is $h \cdot w \cdot \log |\det W|$ for an $h \times w$ feature map.
- **Affine coupling**: as in RealNVP.

Neural ODEs and Continuous Normalising Flows

Continuous Normalising Flows (Chen et al., 2018) replace the discrete composition of flows with a continuous vector field: $\frac{dz_t}{dt} = f_\theta(z_t, t)$, with $z_0 \sim p_0$ and $z_1 \approx p_{\text{data}}$. The log-density evolves via the instantaneous change-of-variables formula:

$$\frac{d \log p(z_t)}{dt} = -\operatorname{tr}\left(\frac{\partial f_\theta}{\partial z_t}\right) \tag{9}$$

Training requires solving both the forward ODE (for samples) and a backward ODE (for log-density changes). The adjoint method reduces memory cost to $O(1)$ in depth, but gradient computation still requires numerical ODE integration, making training slow. This bottleneck motivates the flow matching approach of Chapter 7.

Score-Based Generative Models

Historical and Motivational Context. *Score matching was introduced by Hyvärinen (2005) to train unnormalised energy-based models without computing the partition function. The score function $\nabla_x \log p(x)$ can be estimated by minimising $\mathbb{E}[\|\nabla_x \log p(x) - s_\theta(x)\|^2]$ using integration by parts to avoid the intractable density. Song and Ermon (2019) observed that this objective can be estimated at multiple noise levels simultaneously—each level provides a different "view" of the data distribution—and that combining scores across noise levels via annealed Langevin dynamics produces high-quality samples. The connection to diffusion processes (Song et al., 2021) revealed that score-based models and DDPMs (introduced independently by Ho et al., 2020) are the same*

object viewed from different mathematical perspectives: one continuous and SDE-based, one discrete and VLB-based.

Stein Score Functions and Denoising Score Matching

The Stein score function of a distribution p is $\nabla_x \log p(x)$: the gradient of the log-density. Unlike the density itself, the score does not require knowledge of the normalising constant. For a noisy observation $\tilde{x} = x + \sigma\varepsilon$ with $\varepsilon \sim \mathcal{N}(0, I)$, Vincent (2011) showed that the denoising score matching objective:

$$\mathbb{E}_{x\sim p_{\text{data}},\, \varepsilon\sim\mathcal{N}(0,I)}\left[\|s_\theta(\tilde{x},\sigma) - (-\varepsilon/\sigma)\|^2\right] \tag{10}$$

is equivalent to the original score matching objective up to a constant, making score estimation tractable via noise corruption.

SDEs and the Forward–Reverse Framework

The general framework (Song et al., 2021) defines the forward process as an Itô SDE: $dx = f(x,t)\,dt + g(t)\,dW$ where W is standard Brownian motion. The VPSDE (Variance-Preserving, corresponding to DDPM) uses $f(x,t) = -\frac{1}{2}\beta(t)x$ and $g(t) = \sqrt{\beta(t)}$. The VESDE (Variance-Exploding, corresponding to SMLD) uses $f = 0$ and $g(t) = d\sigma(t)/dt$.

Theorem 4.1 (Probability Flow ODE)**.** *For any SDE of the form $dx = f(x,t)\,dt + g(t)\,dW$, there exists a deterministic ODE—the* probability flow ODE*—with the same marginal distributions $\{p_t\}$:*

$$\frac{dx}{dt} = f(x,t) - \tfrac{1}{2}g(t)^2\nabla_x \log p_t(x) \tag{11}$$

This ODE can be solved with any black-box ODE solver and enables exact likelihood computation via the instantaneous change-of-variables formula.

DDPM, DDIM, and Classifier-Free Guidance

Historical and Motivational Context. *Ho et al. (2020) reframed the score-based generative model as a discrete latent variable model, deriving the training objective as a special case of the variational lower bound on log-likelihood. Their key simplification—predicting noise ε rather than the score or the denoised image—produced a dramatic improvement in sample quality, achieving FID scores competitive with GANs for the first time. Crucially, they observed that the full VLB objective could be simplified: the per-timestep KL terms could be weighted equally (rather than by their analytic magnitudes), producing a simpler objective that trained better in practice. We now derive this simplification carefully.*

DDPM: Full Derivation of the Variational Lower Bound

The forward process adds Gaussian noise over T steps: $q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}\,x_{t-1}, \beta_t I)$. The marginal has a convenient closed form (derived by induction on the convolution of Gaussians):

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}\,x_0,\, (1-\bar{\alpha}_t)I), \quad \bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s) \tag{12}$$

The reverse process $p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t,t), \Sigma_\theta(x_t,t))$ is trained by maximising the ELBO on $\log p_\theta(x_0)$. The full VLB is:

$$-\mathcal{L}_{\text{VLB}} = \mathbb{E}_q\Big[\underbrace{D_{\text{KL}}(q(x_T|x_0)\|p(x_T))}_{\mathcal{L}_T} + \sum_{t=2}^{T}\underbrace{D_{\text{KL}}(q(x_{t-1}|x_t,x_0)\|p_\theta(x_{t-1}|x_t))}_{\mathcal{L}_{t-1}} - \underbrace{\log p_\theta(x_0|x_1)}_{\mathcal{L}_0}\Big] \tag{13}$$

The posterior $q(x_{t-1}|x_t,x_0)$ is tractable: $q(x_{t-1}|x_t,x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t,x_0), \tilde{\beta}_t I)$ where:

$$\tilde{\mu}_t(x_t,x_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t, \quad \tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \tag{14}$$

Each KL term $\mathcal{L}_{t-1} = D_{\text{KL}}(q(x_{t-1}|x_t,x_0)\|p_\theta(x_{t-1}|x_t))$ is a Gaussian KL, computable in closed form. Parameterising μ_θ in terms of a predicted noise $\varepsilon_\theta(x_t,t)$ via $\mu_\theta(x_t,t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon_\theta(x_t,t))$ and fixing $\Sigma_\theta = \sigma_t^2 I$, the KL term simplifies to:

$$\mathcal{L}_{t-1} = \frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\,\mathbb{E}\Big[\|\varepsilon - \varepsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\varepsilon,\, t)\|^2\Big] \tag{15}$$

Ho et al. observed empirically that dropping the time-dependent weighting factor and using the simplified objective

$$\mathcal{L}_{\text{simple}}(\theta) = \mathbb{E}_{t,x_0,\varepsilon}\Big[\|\varepsilon - \varepsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\varepsilon,\, t)\|^2\Big] \tag{16}$$

produces better sample quality in practice. The reweighting upweights small-t (high signal-to-noise) terms relative to the VLB weighting, focusing learning on fine-grained denoising.

DDIM: Deterministic Sampling and the ODE Connection

DDIM (Song et al., 2021) derives a non-Markovian inference process consistent with the same marginals as DDPM but allowing deterministic sampling. The DDIM update is:

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\cdot\frac{x_t - \sqrt{1-\bar{\alpha}_t}\cdot\varepsilon_\theta(x_t,t)}{\sqrt{\bar{\alpha}_t}} + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\cdot\varepsilon_\theta(x_t,t) + \sigma_t\varepsilon \tag{17}$$

Setting $\sigma_t = 0$ gives a deterministic trajectory. The deterministic DDIM corresponds precisely to the Euler discretisation of the probability flow ODE, bridging the discrete and continuous perspectives. DDIM achieves comparable quality to DDPM in 50–100 steps versus DDPM's 1000.

Classifier-Free Guidance

Classifier guidance steers the reverse diffusion toward class y by augmenting the score: $\nabla_x \log p(x_t|y) = \nabla_x \log p_t(x_t) + \gamma\nabla_x \log p(y|x_t)$. Classifier-free guidance (Ho and Salimans, 2021) eliminates the separate classifier by jointly training conditional and unconditional models:

$$\tilde{\varepsilon}_\theta(x_t,t,y) = \varepsilon_\theta(x_t,t,\emptyset) + \gamma\cdot(\varepsilon_\theta(x_t,t,y) - \varepsilon_\theta(x_t,t,\emptyset)) \tag{18}$$

The guidance scale $\gamma > 1$ amplifies the conditional signal, trading diversity for fidelity. The theoretical interpretation: classifier-free guidance samples from the distribution $p(x|y)^\gamma / Z$, a sharpened version of the conditional that concentrates around its modes.

Cross-Reference to Other Parts. This mechanism connects to alignment via reward-weighted regression in Part V: the guidance score $\nabla_x \log p(y|x_t)$ plays the role of a reward signal, steering the reverse process toward high-reward outputs. The same mathematical structure underlies RLHF for language models: the log-ratio $\log(\pi_{\text{fine}}/\pi_{\text{ref}})$ plays the role of the guidance scale.

Consistency Models

Consistency models (Song et al., 2023) address the fundamental tension in diffusion models between sample quality and sampling speed. They learn the consistency function $f : (x_t, t) \mapsto x_0$, which maps any noisy point on the PF ODE trajectory to the trajectory's starting point. The self-consistency property requires: $f(x_{t_1}, t_1) = f(x_{t_2}, t_2)$ for any two times t_1, t_2 on the same trajectory.

The consistency distillation (CD) objective trains f_θ to agree with a target f_{θ^-} (an exponential moving average) on adjacent trajectory points:

$$\mathcal{L}_{\text{CD}}(\theta) = \mathbb{E}\Big[d\Big(f_\theta(x_{t_{n+1}}, t_{n+1}),\ f_{\theta^-}(\hat{x}_{t_n}, t_n)\Big)\Big] \tag{19}$$

where $\hat{x}_{t_n}$ is obtained by one ODE step from $x_{t_{n+1}}$. This enables one-step generation while maintaining competitive quality.

Cross-Reference to Other Parts. Consistency models are related to the contraction mapping theorem (the Banach fixed-point theorem) used in Part V for the Bellman equations: both use the same principle of iterative function approximation under a contraction metric. The exponential moving average target f_{θ^-} is also used in DQN target networks (Part V) for the same purpose—stabilising a self-referential fixed-point objective.

Flow Matching and Optimal Transport

Historical and Motivational Context. *Flow matching was introduced independently by Lipman et al. (2022) as "flow matching" and by Liu et al. (2022) as "rectified flows". The key conceptual advance over CNFs was to replace the ODE-solver-based training with direct regression on a conditional vector field: rather than integrating forward and backward through an ODE to compute the likelihood, simply regress the network output against the velocity of a known conditional path. This is analogous to the transition from the variational lower bound to score matching: a loss function that avoids computing the intractable marginal quantity (the ODE trajectory) by conditioning on known endpoints. The optimal transport variant (OT-FM) further showed that the choice of coupling between noise and data profoundly affects the geometry of the learned vector field, with the OT coupling producing the straightest possible trajectories.*

Conditional Flow Matching

Let $p_0 = \mathcal{N}(0, I)$ and $p_1 = p_{\text{data}}$. Flow matching seeks a vector field $u_t : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$ whose generated flow ϕ_t satisfies $[\phi_t]_{\#} p_0 = p_t$ for all t. The marginal flow matching loss is:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t, x \sim p_t}\left[\|v_\theta(x_t, t) - u_t(x_t)\|^2\right] \tag{20}$$

Since the marginal vector field u_t is intractable, Conditional Flow Matching (CFM) conditions on the data point x_1: for the simple linear interpolant $\phi_t(x_0, x_1) = (1-t)x_0 + t x_1$, the conditional velocity is simply $(x_1 - x_0)$, giving:

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t, x_0, x_1}\left[\|v_\theta(\phi_t(x_0, x_1), t) - (x_1 - x_0)\|^2\right] \tag{21}$$

The key theorem is that $\mathcal{L}_{\text{CFM}}$ and $\mathcal{L}_{\text{FM}}$ have identical gradients despite conditioning on x_1.

OT-Flow Matching and Straight Trajectories

The optimal transport coupling $\pi^* = \arg\min_\pi \mathbb{E}_{(x_0,x_1)\sim\pi}[\|x_0 - x_1\|^2]$ minimises the expected transport cost, producing straighter flow trajectories.

Theorem 7.1 (Straightness and NFE Efficiency)**.** *For a flow generated by interpolant $\phi_t(x_0, x_1)$, define the straightness $S = 1 - \mathbb{E}[\|\phi_t(x_0,x_1) - (t \cdot x_1 + (1-t)\cdot x_0)\|^2]/\mathbb{E}[\|x_1 - x_0\|^2]$. The OT coupling maximises straightness among all couplings with the same marginals. A flow with curvature κ requires $O(\kappa/\varepsilon)$ steps for error ε, while a straight flow requires $O(1/\varepsilon)$.*

Stochastic Interpolants and the Broader Framework

Stochastic interpolants (Albergo and Vanden-Eijnden, 2023) unify diffusion models and flow matching: $I_t(x_0, x_1) = \alpha(t)x_0 + \beta(t)x_1 + \gamma(t)\varepsilon$ where $\varepsilon \sim \mathcal{N}(0, I)$. Setting $\gamma = 0$ recovers deterministic flow matching; setting $\alpha(t) = \sqrt{\bar{\alpha}_t}$ and $\beta(t) = \sqrt{1 - \bar{\alpha}_t}$ recovers DDPM-type diffusion.

> **Cross-Reference to Other Parts.** Flow matching connects directly to the information geometry of Part I. The vector field u_t defines a geodesic in the space of probability measures equipped with the W_2 metric—the Benamou–Brenier formulation of optimal transport. The Fisher information metric on the statistical manifold (Part I) and the Wasserstein metric on the space of distributions are related by Otto's calculus (2001), which identifies the L^2 Wasserstein space as an infinite-dimensional Riemannian manifold and the continuity equation $\partial_t p_t + \nabla \cdot (p_t u_t) = 0$ as its geodesic equation.

Latent Diffusion Models and VQ-VAE

VQ-VAE: Vector Quantisation in Latent Space

The Vector Quantised VAE (van den Oord et al., 2017) learns a discrete latent representation. An encoder maps x to a continuous embedding z_e, which is quantised to the nearest codebook entry: $z_q = \arg\min_{e_k \in \mathcal{C}} \|z_e - e_k\|_2$. The loss is:

$$\mathcal{L}_{\text{VQVAE}} = \|x - D(z_q)\|^2 + \|sg[z_e] - z_q\|^2 + \beta\|z_e - sg[z_q]\|^2 \tag{22}$$

where $sg[\cdot]$ is the stop-gradient operator. The straight-through gradient $\nabla_{z_e}\mathcal{L} \approx \nabla_{z_q}\mathcal{L}$ allows approximate gradient flow through the non-differentiable arg min operation.

Latent Diffusion Models and Cross-Attention Conditioning

LDM (Rombach et al., 2022) trains a KL-regularised autoencoder to compress images by a factor of 4–8 in each spatial dimension, then trains a diffusion model on the compressed latent code. The computational savings are f^4 for an f-fold spatial compression. Conditioning is via cross-attention:

$$\text{Attention}(Q,K,V) = \text{softmax}(QK^T/\sqrt{d})\cdot V, \quad Q = W_Q\cdot\varphi(z_t),\ K = V = W_{K,V}\cdot\tau_\theta(y) \tag{23}$$

where $\tau_\theta(y)$ is a domain-specific encoder (e.g., CLIP text encoder).

Cross-Reference to Other Parts. The VQ-VAE's discrete latent space connects to the information-bottleneck perspective of Part III, where compression is measured by the minimum description length of the representation. The straight-through estimator is an application of the same gradient heuristic used in binary stochastic networks; its theoretical justification connects to the score function estimators studied in Part III.

Energy-Based Models and Modern Hopfield Networks

Historical and Motivational Context. *Energy-based models (Gibbs, 1902; Boltzmann, 1884) predate neural networks: the Boltzmann distribution $p(x) \propto \exp(-E(x)/T)$ is the fundamental object of statistical mechanics. Their application to learning (Ackley, Hinton, and Sejnowski, 1985) led to Boltzmann Machines, but tractable training required restricting the architecture (RBMs). LeCun et al. (2006) proposed a general EBM framework for machine learning. The modern revival of EBMs (Du and Mordatch, 2019; Grathwohl et al., 2020) combines deep energy functions with improved MCMC methods (SGLD, persistent CD) and connects EBMs to diffusion models: the score of an EBM is $-\nabla E(x)$, so a score-based diffusion model is precisely an EBM whose energy function is trained via denoising score matching at multiple noise levels.*

The EBM Framework and Contrastive Divergence

An EBM defines $p_\theta(x) = \exp(-E_\theta(x))/Z_\theta$ where $Z_\theta = \int \exp(-E_\theta(x))\,dx$. The maximum likelihood gradient requires: $\nabla_\theta \log p_\theta(x) = -\nabla_\theta E_\theta(x) + \mathbb{E}_{p_\theta}[\nabla_\theta E_\theta(x)]$. Contrastive divergence (Hinton, 2002) approximates the model expectation by running MCMC for k steps from data (CD-k), giving a biased but computationally efficient gradient estimate.

Modern Hopfield Networks and the Attention Connection

Classical Hopfield networks (1982) store memory patterns $\{\xi^\mu\}$ as attractors of the energy $E(x) = -\frac{1}{2}x^TWx$ with $W = \sum_\mu \xi^\mu(\xi^\mu)^T$. Memory capacity is limited to $O(n)$ patterns. Modern Hopfield Networks (Ramsauer et al., 2021) replace the quadratic energy with an exponential interaction:

$$E(x,\Xi) = -\,\text{lse}(\beta,\Xi^Tx) + \tfrac{1}{2}x^Tx + \beta^{-1}\log M + \tfrac{1}{2}\beta^{-1}\max_\mu \|\xi^\mu\|^2 \tag{24}$$

The update rule $x^{\text{new}} = \Xi\,\text{softmax}(\beta \Xi^T x)$ is precisely the attention mechanism. This reveals that transformer attention layers can be interpreted as one step of a Modern Hopfield update, and that the exponential energy gives capacity $2^{n/2}$ patterns—exponential in dimension—versus $O(n)$ for classical Hopfield.

Theorem 9.1 (Exponential Storage Capacity of Modern Hopfield Networks)**.** *Let $\{\xi^\mu\}_{\mu=1}^M$ be M patterns with $\|\xi^\mu\| \le 1$. The Modern Hopfield network retrieves patterns with exponentially small error provided $M \le \exp(\alpha n)$ for some $\alpha < \frac{1}{2}\beta^2 \cdot (\text{min separation})^2$. The retrieval error is bounded by $\exp(-\beta^2 \cdot (\text{separation})^2/2) \cdot n$.*

Self-Supervised Learning

Historical and Motivational Context. *Self-supervised learning has been practised implicitly since autoencoders (Rumelhart et al., 1986), but the modern era begins with Word2Vec (Mikolov et al., 2013), which showed that predicting word context produces representations that encode semantic meaning. BERT (Devlin et al., 2019) extended this to masked language modelling; SimCLR (Chen et al., 2020) demonstrated that contrastive objectives applied to augmented image views produce representations competitive with supervised learning on ImageNet. The theoretical question—why does self-supervised learning produce useful representations?—was first addressed rigorously by Wang and Isola (2020), who decomposed the InfoNCE loss into alignment and uniformity terms and proved that optimal representations are uniform on the sphere with positive pairs co-located. This provided the first rigorous geometric characterisation of what contrastive learning is optimising.*

Contrastive Learning: SimCLR and the InfoNCE Objective

SimCLR (Chen et al., 2020) creates two augmented views x, x^+ of the same image. The NT-Xent (InfoNCE) loss for a batch of N pairs:

$$\mathcal{L} = -\frac{1}{N}\sum_i \log \frac{\exp(\text{sim}(z_i, z_i^+)/\tau)}{\sum_{j \ne i} \exp(\text{sim}(z_i, z_j)/\tau)} \tag{25}$$

where $z_i = g(f(x_i))$ and $\text{sim}(u, v) = u^T v/(\|u\|\|v\|)$. The loss decomposes into:

$$\begin{aligned} \mathcal{L}_{\text{align}} &= \mathbb{E}_{(x,x^+)}[\|f(x) - f(x^+)\|^2] \\ \mathcal{L}_{\text{uniform}} &= \log \mathbb{E}_{x,y}[\exp(-2\|f(x) - f(y)\|^2)] \end{aligned} \tag{26}$$

Optimal representations are uniform on the sphere while positive pairs are co-located.

BYOL: Self-Distillation Without Negative Pairs

BYOL (Grill et al., 2020) trains an online network to predict the output of a target network (EMA of online) without negative pairs: $\mathcal{L}_{\text{BYOL}} = \|q_\theta(z_{\text{online}}) - sg[z_{\text{target}}]\|_2^2$. The absence of negative pairs raises the collapse question. Tian et al. (2021) show that BatchNorm introduces an implicit negative pair mechanism via cross-batch statistics, preventing collapse.

Barlow Twins and Information-Theoretic Redundancy Reduction

Barlow Twins (Zbontar et al., 2021) enforces decorrelation between feature dimensions:

$$\mathcal{L}_{\text{BT}} = \sum_i (1 - C_{ii})^2 + \lambda \sum_i \sum_{j \neq i} C_{ij}^2, \quad C_{ij} = \frac{\sum_b z_b^i z_b^j}{\|z^i\| \|z^j\|} \tag{27}$$

The first term pushes diagonal toward 1 (invariance to augmentation); the second pushes off-diagonal toward 0 (redundancy reduction). This is a lower bound on the mutual information between the two views under the constraint that the feature distribution is approximately white.

Masked Autoencoders

MAE (He et al., 2022) masks 75% of image patches and trains a ViT encoder–decoder to reconstruct the masked pixels. Key design choices: (1) high masking ratio creates a harder, information-rich task; (2) encoder operates only on visible tokens, dramatically reducing compute; (3) shallow decoder processes the full sequence. MAE demonstrates that visual representations of competitive quality can be learned purely from reconstruction, without contrastive objectives or explicit augmentation policies.

> **Cross-Reference to Other Parts.** Masked autoencoders are the vision analogue of BERT's masked language modelling (Part II). The ViT encoder used in MAE is the same architecture analysed in Part II for its attention properties. The information-theoretic interpretation—high masking creates a high-compression pretext task—connects to the MDL principle of Part III: the model learns to compress the visible context sufficiently to reconstruct the masked content.

Normalising Flows: Advanced Architectures

Autoregressive Flows and the MADE Architecture

Masked Autoregressive Flow (MAF, Papamakarios et al., 2017) constructs a flow using the autoregressive factorisation: $p(x) = \prod_i p(x_i | x_{1:i-1})$. Each conditional is parameterised as a Gaussian: $p(x_i | x_{1:i-1}) = \mathcal{N}(x_i; \mu_i(x_{1:i-1}), \sigma_i^2(x_{1:i-1}))$. The log-likelihood is: $\log p(x) = \sum_i [-\frac{1}{2}\log(2\pi\sigma_i^2) - \frac{(x_i - \mu_i)^2}{2\sigma_i^2}]$. The Jacobian is triangular (each x_i depends only on $x_{1:i-1}$), so $\log|\det J| = -\sum_i \log \sigma_i(x_{1:i-1})$ in $O(d)$.

MADE (Germain et al., 2015) implements the autoregressive constraint via masked feedforward networks: weights between layer l and layer $l+1$ are masked so that unit i in layer $l+1$ receives input only from units $j \leq i$ in layer l. This enables a single network to compute all conditionals μ_i, σ_i simultaneously in a single forward pass.

Limitation. MAF is fast to evaluate likelihood but slow to sample: generating x_i requires $\mu_i(x_{1:i-1})$ and $\sigma_i(x_{1:i-1})$, so sampling is sequential ($O(d)$ network evaluations). IAF (Inverse Autoregressive Flow, Kingma et al., 2016) inverts the parameterisation to make sampling fast at the cost of slow likelihood evaluation.

Neural Spline Flows

Neural Spline Flows (Durkan et al., 2019) replace the affine coupling ($y = \exp(s)x + t$) with a monotone rational-quadratic spline g_K: $y = g_K(x; w, h, d)$ where w are widths, h are heights, and d are derivatives at K knots. The spline is piecewise rational-quadratic, ensuring:

- Monotone (hence invertible)
- Analytically invertible (closed form g_K^{-1})
- Flexible (can approximate any monotone function with K knots)

The log-determinant is $\sum_i \log g_K'(x_i)$ (product of spline derivatives), computable in $O(d)$. NSF significantly improves density estimation over affine coupling layers, especially on multimodal distributions.

Continuous Normalising Flows: The FFJORD Algorithm

FFJORD (Grathwohl et al., 2019) avoids computing the full Jacobian of the ODE by using Hutchinson's estimator for the trace: $\log|\det J_f| = \int_0^1 \operatorname{tr}\left(\frac{\partial f_\theta}{\partial z_t}\right) dt \approx \int_0^1 v^T \frac{\partial f_\theta}{\partial z_t} v\, dt$ where $v \sim \mathcal{N}(0, I)$ is a random vector. This reduces the per-step cost from $O(d^2)$ (full Jacobian) to $O(d)$ (vector-Jacobian product), enabling CNFs to scale to high-dimensional data.

Diffusion Models: Advanced Topics

Improved Noise Schedules: Cosine and Perceptual Weighting

The original DDPM linear noise schedule leads to suboptimal sample quality because early timesteps ($t \approx T$, high noise) are already close to $\mathcal{N}(0, I)$ and provide little training signal, while the information-rich low-noise regime ($t \approx 0$) receives equal weight. Improved DDPM (Nichol and Dhariwal, 2021) introduces:

Cosine schedule: $\bar{\alpha}_t = f(t)/f(0)$ where $f(t) = \cos^2(\pi t/(2T + 0.008\pi))$. This provides a more uniform SNR across timesteps, reducing the fraction of training steps wasted on near-Gaussian noise.

Perceptual weighting: $\lambda(t) = \sigma_t^{-2}$ in the diffusion objective upweights low-noise timesteps (where the signal has high perceptual quality) relative to high-noise timesteps. This can be modified to weight timesteps by their contribution to the final sample quality.

Conditional Generation: Diffusion for Text-to-Image

Stable Diffusion (Rombach et al., 2022) combines latent diffusion with CLIP text conditioning:

1. A variational autoencoder encodes 512×512 images to $64 \times 64 \times 4$ latents.
2. A CLIP text encoder maps the text prompt to a sequence of embeddings.
3. The U-Net denoises the latent while attending to text embeddings via cross-attention

at every resolution level.

4. Classifier-free guidance is applied during inference.

The U-Net architecture uses ResNet blocks for spatial processing and self-attention for long-range dependencies, with cross-attention to text: $\text{Attn}(Q, K, V) = \text{softmax}(QK^T/\sqrt{d})V$, $Q = W_Q\varphi(z_t)$, $K = W_K\tau(y)$, $V = W_V\tau(y)$, where $\tau(y)$ is the CLIP text embedding and $\varphi(z_t)$ is a flattened spatial feature map.

Diffusion Transformers (DiT)

DiT (Peebles and Xie, 2023) replaces the U-Net backbone with a pure transformer, treating the noisy latent as a sequence of patches. Each 2×2 patch is a token, and the transformer processes all patches in parallel. Time and class conditioning are incorporated via adaptive layer norm: $h' = (1 + \gamma)h + \beta$ where $(\gamma, \beta) = \text{MLP}(t, c)$.

DiT scales better than U-Net with compute: FID on ImageNet decreases approximately as $O(\text{GFlops}^{-0.6})$ for DiT but saturates for U-Net-based architectures. This motivated the shift from U-Net to transformer backbones in Stable Diffusion 3 and subsequent text-to-image models.

Cross-Reference to Other Parts. The DiT architecture is an instance of the ViT (Vision Transformer) analysed in Part II: both treat images as sequences of patches processed by a transformer. The connection between DiT and the DDPM framework (this part, Chapter 5) is that DiT simply replaces the spatial feature extractor (U-Net) with a sequence model (transformer), while keeping the denoising objective unchanged. The scaling laws for DiT (Part VI, Chapter 4) follow the same power-law in compute as language model scaling.

Evaluating Generative Models

Fréchet Inception Distance and Precision-Recall

The Fréchet Inception Distance (FID, Heusel et al., 2017):

$$\text{FID} = \|\mu_r - \mu_g\|_2^2 + \text{tr}\big(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}\big) \tag{28}$$

is the W_2 distance between two Gaussians fitted to the Inception-v3 feature distributions of real (r) and generated (g) images. Lower FID indicates better distributional match.

Derivation. The W_2 distance between $\mathcal{N}(\mu_1, \Sigma_1)$ and $\mathcal{N}(\mu_2, \Sigma_2)$ is: $W_2^2 = \|\mu_1 - \mu_2\|^2 + \text{tr}(\Sigma_1 + \Sigma_2 - 2(\Sigma_1^{1/2}\Sigma_2\Sigma_1^{1/2})^{1/2})$, which reduces to the FID formula when $\Sigma_1 = \Sigma_r$ and $\Sigma_2 = \Sigma_g$.

Limitations. FID aggregates fidelity and diversity into a single scalar. Precision measures what fraction of generated samples are realistic (near the real data manifold); recall measures what fraction of the real data manifold is covered by generated samples. These are estimated via k-NN in feature space (Kynkäänniemi et al., 2019).

Negative Log-Likelihood and Bits per Dimension

For flow models and diffusion models with tractable likelihoods, the bits per dimension (BPD) is the standard metric: $\text{BPD} = -\log_2 p_\theta(x)/d$ where d is the dimensionality. Bits per dimension is an absolute measure of compression efficiency: a BPD of 3.5 means the model uses 3.5 bits to encode each pixel, compared to 8 bits for raw pixels. State-of-the-art generative models on CIFAR-10 achieve BPD ≈ 2.55 (autoregressive) to 2.74 (diffusion), close to the estimated entropy of natural images.

Score-Based Models: Convergence Analysis and Noise Schedule Design

Historical and Motivational Context. *The theoretical analysis of score-based generative models—when do they converge, how fast, and what controls the error—lagged behind empirical developments. Early work (Song and Ermon, 2019) demonstrated empirically that using multiple noise levels dramatically improved sample quality, but provided no convergence guarantees. The rigorous theory was developed concurrently by several groups (De Bortoli et al., 2021; Chen et al., 2022; Lee et al., 2022), establishing that the discretisation error of the reverse SDE depends on the score estimation error and the noise schedule in quantifiable ways. The connection to Langevin dynamics and MCMC theory provided the conceptual bridge between score matching and the rich literature on convergence of gradient-based sampling.*

Denoising Score Matching: Equivalence to Score Matching

The original score matching objective (Hyvärinen, 2005) requires computing $\nabla_x \log p(x)$ of the data density, which involves second-order information:

$$J_{\text{SM}}(\theta) = \mathbb{E}_{p(x)}\left[\frac{1}{2}\|s_\theta(x)\|^2 + \nabla_x \cdot s_\theta(x)\right] \tag{29}$$

The divergence term $\nabla_x \cdot s_\theta(x)$ requires computing the trace of the Jacobian, scaling as $O(d)$ via Hutchinson's estimator (a random vector $v \sim \mathcal{N}(0, I)$: $\nabla_x \cdot s = \mathbb{E}_v[v^T \nabla_x s\, v]$) but still computationally expensive.

Denoising score matching (Vincent, 2011) avoids the Jacobian entirely. For a noisy observation $\tilde{x} = x + \sigma\varepsilon$ with $\varepsilon \sim \mathcal{N}(0, I)$:

$$\begin{aligned} J_{\text{DSM}}(\theta) &= \mathbb{E}_{p(x),\varepsilon}\left[\|s_\theta(\tilde{x}) - \nabla_{\tilde{x}} \log p_\sigma(\tilde{x}|x)\|^2\right] \\ &= \mathbb{E}_{p(x),\varepsilon}\left[\|s_\theta(\tilde{x}) - (-\varepsilon/\sigma)\|^2\right] \end{aligned} \tag{30}$$

The equivalence $J_{\text{DSM}} = J_{\text{SM}} + C$ (for constant C) means that denoising score matching and score matching have the same minimiser, but DSM is computationally tractable: no Jacobian, just mean-squared error between the network output and the known noise direction $-\varepsilon/\sigma$.

The Noise Schedule: Variance-Preserving vs Variance-Exploding

The VPSDE uses a noise schedule $\beta(t) : [0, 1] \to \mathbb{R}_{>0}$ such that:

$$dx = -\frac{1}{2}\beta(t)x\, dt + \sqrt{\beta(t)}\, dW \tag{31}$$

The marginal $p_t(x_t|x_0) = \mathcal{N}(\sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I)$ where $\bar{\alpha}_t = \exp(-\int_0^t \beta(s)ds)$. As $t \to T$: $\bar{\alpha}_T \to 0$, so $p_T(x_T) \approx \mathcal{N}(0, I)$ regardless of x_0. This is the *variance-preserving* property: $\text{Var}[x_t] = \bar{\alpha}_t \,\text{Var}[x_0] + (1-\bar{\alpha}_t) \approx 1$ for unit-variance data.

Common schedule choices:

- **Linear:** $\beta(t) = \beta_{\min} + t(\beta_{\max} - \beta_{\min})$. Simple but not optimal.
- **Cosine:** $\bar{\alpha}_t = \cos^2(\pi t/(2+0.008\pi))$. Better near $t = 0$ (avoids blurring fine details too early).
- **Sigmoid:** $\bar{\alpha}_t = \sigma(-c(t-0.5))$. Flexible parameterisation.

Convergence Analysis of the Reverse SDE

The reverse-time SDE generates samples from p_0 by solving: $dx = [-f(x,t) + g(t)^2 \nabla_x \log p_t(x)]\,dt + g(t)\,d\bar{W}$ from $x_T \sim p_T \approx \mathcal{N}(0, I)$.

In practice, the score $\nabla_x \log p_t(x)$ is replaced by the learned $s_\theta(x,t)$. The generation error decomposes as:

$$\text{TV}(p_0, \hat{p}_0) \leq \underbrace{C_1 T\, \varepsilon^2_{\text{score}}}_{\text{score estimation error}} + \underbrace{C_2\, N^{-1} \sum_k h_k^2}_{\text{discretisation error}} \tag{32}$$

where $\varepsilon^2_{\text{score}} = \mathbb{E}_{t,x_t}[\|s_\theta(x_t,t) - \nabla_{x_t} \log p_t(x_t)\|^2]$ is the score error, N is the number of discretisation steps, and h_k are the step sizes. This bound motivates (1) training to minimise the score error uniformly over all noise levels, and (2) using adaptive step sizes (smaller steps near $t = 0$ where the score changes rapidly).

GAN Training: Advanced Stabilisation Techniques

Spectral Normalisation and the Lipschitz Constant

Spectral normalisation (Miyato et al., 2018) enforces the 1-Lipschitz constraint on the discriminator by normalising each weight matrix by its spectral norm $\sigma_{\max}(W)$. The power iteration algorithm estimates $\sigma_{\max}$ efficiently:

$$\begin{aligned} \tilde{v} &= W^T\tilde{u}/\|W^T\tilde{u}\|, \quad \tilde{u} = W\tilde{v}/\|W\tilde{v}\| \\ \sigma_{\max} &\approx \tilde{u}^T W \tilde{v} \end{aligned} \tag{33}$$

requiring only two matrix-vector products per update step.

For a network with L layers, $\|f\|_L \leq \prod_\ell \sigma_{\max}(W_\ell)$, so normalising each layer to unit spectral norm bounds the global Lipschitz constant by 1. In practice, only the discriminator is spectrally normalised; the generator is unconstrained (its Lipschitz constant is irrelevant for the WGAN objective).

Progressive Growing and Multi-Scale Training

Progressive GAN (Karras et al., 2018) addresses the difficulty of training high-resolution generators by progressively growing both generator and discriminator: starting at 4×4

resolution and doubling each time the training has stabilised. When a new resolution level is added, it is faded in gradually using a weight α that increases from 0 to 1:

$$x_{\text{in}} = (1-\alpha)\,\text{upsample}(h_{k-1}) + \alpha\, h_k \tag{34}$$

This smooth transition prevents the training from being disrupted by the sudden addition of new layers.

FID, IS, and Precision-Recall

The Fréchet Inception Distance (FID, Heusel et al., 2017) measures sample quality by comparing the Gaussian statistics of Inception-v3 activations:

$$\text{FID} = \|\mu_r - \mu_g\|^2 + \text{tr}\big(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}\big) \tag{35}$$

where (μ_r, Σ_r) and (μ_g, Σ_g) are the mean and covariance of real and generated features. FID is the squared Wasserstein-2 distance between two Gaussians fitted to the feature distributions—a closed-form approximation of the true distribution distance.

Limitations. FID aggregates fidelity and diversity into a single number. Precision–Recall (Kynkäänniemi et al., 2019) separates these: precision measures what fraction of generated samples fall in the real manifold (fidelity); recall measures what fraction of the real manifold is covered by generated samples (diversity). A mode-collapsed GAN has high precision but low recall.

The Geometric Unity of Generative Methods

Optimal Transport as the Unifying Framework

All generative methods can be understood as solving the optimal transport problem: finding the most efficient coupling between the simple source distribution $p_z = \mathcal{N}(0, I)$ and the complex target p_{data}.

The Benamou–Brenier formulation (2000) expresses the W_2 distance as:

$$W_2^2(p_0, p_1) = \inf_{\rho, v} \int_0^1 \int \|v(x,t)\|^2 \rho(x,t)\, dx\, dt \tag{36}$$

subject to $\partial_t \rho + \nabla \cdot (\rho v) = 0$ (continuity equation). This is a dynamical formulation: the W_2 distance is the minimum kinetic energy of a flow transporting p_0 to p_1.

Each generative method corresponds to a different solution of this problem:

- **Flow matching with OT coupling**: solves exactly, with straight trajectories.
- **DDPM**: adds stochastic noise to the transport path (not OT-optimal, but enables score estimation).
- **GAN**: solves adversarially without a transport path, using the discriminator as a critic.
- **VAE**: solves approximately via a variational lower bound.

The Role of the Data Manifold

The manifold hypothesis—that high-dimensional data concentrates near a low-dimensional manifold $\mathcal{M} \subset \mathbb{R}^D$—profoundly affects generative models. When p_{data} is supported on a manifold of dimension $d \ll D$:

- The score $\nabla_x \log p_{\text{data}}(x)$ is undefined off the manifold.
- This is why score-based models require adding noise: $p_\sigma(x) = \int p(y)\mathcal{N}(x; y, \sigma^2 I)\, dy$ smooths the distribution off the manifold, making the score well-defined.
- The diffusion forward process can be understood as progressively inflating the data manifold until it fills $\mathbb{R}^D$, at which point the reverse process contracts it back.

Cross-Reference to Other Parts. The manifold hypothesis is the bridge between the generative methods of this part and the geometric foundations of Part I. The intrinsic dimension of the data manifold (measurable by the local PCA spectrum of the data covariance) determines the effective dimension of the latent space in VAEs and the effective rank of the score function. The Whitney embedding theorem guarantees that a manifold of dimension d can be embedded in $\mathbb{R}^{2d+1}$, providing a lower bound on the latent dimension needed for faithful reconstruction.

Discrete Generative Models and Masked Diffusion

While this part has focused on continuous generative models (images, audio, video), language models are generative models over discrete sequences. The connection between discrete and continuous generation is provided by *masked diffusion* (Austin et al., 2021; Sahoo et al., 2024), which defines a forward process that progressively masks tokens:

$$q(x_t|x_0) = \prod_i \left[(1 - m_t)\delta_{x_0^i} + m_t \delta_{[MASK]}\right] \tag{37}$$

where m_t is the masking rate at time t. The reverse process denoises masked tokens, predicting the original token from the masked context. MDLM (Sahoo et al., 2024) shows that masked diffusion achieves competitive perplexity to autoregressive models on language benchmarks, with the advantage of parallel decoding—sampling all tokens simultaneously from the joint reverse process rather than sequentially.

Advanced Topics in Latent Generation

Stable Diffusion 3 and Multi-Flow Architecture

Stable Diffusion 3 (Esser et al., 2024) applies flow matching to the latent space of a VAE, combining four streams:

1. A variational autoencoder compresses 512×512 RGB images to $64 \times 64 \times 16$ latents.
2. A flow matching network (the DiT, Diffusion Transformer) operates on the latent space.
3. Text conditioning via CLIP and T5 text encoders.

4. A rectified flow schedule with timestep shifting.

The DiT (Peebles and Xie, 2023) replaces the U-Net with a pure transformer architecture, treating the noisy latent as a sequence of patches. The time conditioning and class conditioning are incorporated via adaptive layer norm: $\text{AdaLN}(h, t, c) = (1+\gamma(t,c))h + \beta(t,c)$, where γ and β are linear projections of the time and class embeddings.

Video Diffusion and Spatio-Temporal Consistency

Video generation extends image diffusion to the temporal dimension. The key challenge is maintaining temporal consistency: adjacent frames must depict the same scene with smooth motion, not independent samples.

Video Diffusion Models (Ho et al., 2022) extend the U-Net with temporal attention layers: at each resolution level, a spatial attention layer is followed by a temporal attention layer that attends across frames:

$$h'_t = \text{SpatialAttn}(h_t), \quad h''_t = \text{TemporalAttn}(\{h'_1, \dots, h'_T\}) \tag{38}$$

The temporal attention uses relative position encodings that decrease with frame distance, encoding the inductive bias that nearby frames are more related. Sora (OpenAI, 2024) extends this to video of arbitrary length and resolution using a space-time transformer that treats video frames as a sequence of patches in both space and time.

> **Cross-Reference to Other Parts.** Video diffusion connects to the optimal transport perspective of this part: temporal consistency requires that the distribution of frames at each time t is a smooth interpolation along a path in the space of image distributions—analogous to the flow matching perspective on image generation, but now with the additional constraint that the path must be physically plausible. The Benamou–Brenier kinetic energy formulation naturally extends to video: the "transport cost" is the sum over time of the optical flow magnitudes, and video generation corresponds to finding the minimum-cost spatiotemporal flow.

Variational Inference: Beyond the Mean-Field Approximation

Normalising Flow Posteriors and Hierarchical Inference

The mean-field VAE assumes a diagonal Gaussian posterior $q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \text{diag}(\sigma^2_\phi(x)))$. This restricts the posterior to axis-aligned ellipsoids, missing correlations between latent dimensions. Flow-based posteriors replace this with a more expressive family:

$$q_\phi(z_K|x) = q_\phi(z_0|x) \prod_{k=1}^{K} \left| \det \frac{\partial f_k}{\partial z_{k-1}} \right|^{-1} \tag{39}$$

where $z_0 \sim q_0$ (a simple Gaussian) and $z_K = f_K \circ \cdots \circ f_1(z_0)$ is the result of K normalising flow steps. This enables the posterior to be multimodal, non-axis-aligned, and arbitrarily complex, while maintaining tractable ELBO computation.

Hierarchical VAEs (NVAE, VDVAE) further improve the posterior by introducing multiple levels of latent variables: $q_\phi(z_1, \dots, z_L|x) = \prod_\ell q_\phi(z_\ell|z_{<\ell}, x)$ with the ELBO telescoping as: $\mathcal{L} = \mathbb{E}[\log p_\theta(x|z_1)] - \sum_\ell D_{KL}(q_\phi(z_\ell|z_{<\ell}, x) \| p_\theta(z_\ell|z_{<\ell}))$.

Importance Weighted Autoencoders

IWAE (Burda, Grosse, Salakhutdinov, 2016) tightens the ELBO by using $k > 1$ importance-weighted samples:

$$\mathcal{L}^k_{\text{IWAE}} = \mathbb{E}_{z_1,\ldots,z_k \sim q_\phi}\left[\log \frac{1}{k}\sum_{i=1}^{k} \frac{p_\theta(x, z_i)}{q_\phi(z_i|x)}\right] \tag{40}$$

By Jensen's inequality, $\mathcal{L}^k_{\text{IWAE}} \geq \mathcal{L}^{k-1}_{\text{IWAE}} \geq \mathcal{L}^1_{\text{IWAE}} = \text{ELBO}$, and $\mathcal{L}^k \to \log p_\theta(x)$ as $k \to \infty$. The IWAE estimator is an importance weighted estimate of the marginal log-likelihood using $z_i \sim q_\phi$ as proposal and $p_\theta(x, z_i)/q_\phi(z_i|x)$ as the importance weight. The bound tightens with more samples because the maximum of k samples approaches the true value.

Connections: Diffusion, Score Matching, and Denoising Autoencoders

The Score Function as a Denoising Direction

Tweedie's formula (Robbins, 1956) provides a direct connection between the score function and the optimal denoiser. For $\tilde{x} = x + \sigma\varepsilon$, $\varepsilon \sim \mathcal{N}(0, I)$, the posterior mean is:

$$\mathbb{E}[x|\tilde{x}] = \tilde{x} + \sigma^2 \nabla_{\tilde{x}} \log p_\sigma(\tilde{x}) \tag{41}$$

This is Tweedie's formula: the score function at $\tilde{x}$ points from $\tilde{x}$ toward the posterior mean $\mathbb{E}[x|\tilde{x}]$. The score function is exactly the normalised direction in which to denoise the noisy observation.

This gives a clean interpretation of DDPM: the network $\varepsilon_\theta(x_t, t)$ predicts the noise ε added to x_0. By Tweedie's formula: $\hat{x}_0 = x_t - \sqrt{1-\bar{\alpha}_t}\,\varepsilon_\theta(x_t, t)/\sqrt{\bar{\alpha}_t}$. The score is then $\nabla_{x_t} \log p_t(x_t) = -\varepsilon_\theta(x_t, t)/\sqrt{1-\bar{\alpha}_t}$.

Elucidating the Design Space of Diffusion Models

EDM (Karras et al., 2022) provides a systematic analysis of the design space for diffusion models by reparameterising the denoising network as: $D_\theta(x;\sigma) = c_{\text{skip}}(\sigma)x + c_{\text{out}}(\sigma)F_\theta(c_{\text{in}}(\sigma)x; c_{\text{noise}}(\sigma))$ where the scaling factors $c_{\text{skip}}, c_{\text{out}}, c_{\text{in}}, c_{\text{noise}}$ are functions of the noise level σ chosen to equalise the magnitude of the network input and output across noise levels.

The optimal scalings minimise the expected squared magnitude of F_θ over the training distribution:

$$\begin{aligned} c_{\text{in}}(\sigma) &= 1/\sqrt{\sigma^2 + \sigma^2_{\text{data}}} \\ c_{\text{skip}}(\sigma) &= \sigma^2_{\text{data}}/(\sigma^2 + \sigma^2_{\text{data}}) \\ c_{\text{out}}(\sigma) &= \sigma \cdot \sigma_{\text{data}}/\sqrt{\sigma^2 + \sigma^2_{\text{data}}} \end{aligned} \tag{42}$$

EDM with these scalings achieves FID 1.79 on CIFAR-10 (state of the art among diffusion models), demonstrating that careful normalisation throughout training is as important as architecture choice.

Worked Derivations: Connecting Score, Noise, and Denoised Image

From Noise Prediction to Score: Three Equivalent Parameterisations

The denoising network at noise level σ can be parameterised in three equivalent ways, each with different numerical properties:

1. Noise prediction (ε-parameterisation, Ho et al. 2020): $\varepsilon_\theta(x_t, t) \approx \varepsilon$, predicting the noise added to x_0. Score: $s_\theta(x_t, t) = -\varepsilon_\theta(x_t, t)/\sqrt{1-\bar{\alpha}_t}$.

2. Score prediction (s-parameterisation, Song et al. 2021): $s_\theta(x_t, t) \approx \nabla_{x_t} \log p_t(x_t)$. Noise: $\varepsilon_\theta = -\sqrt{1-\bar{\alpha}_t}\, s_\theta(x_t, t)$.

3. Clean image prediction (x_0-parameterisation): $\hat{x}_\theta(x_t, t) \approx x_0$. By Tweedie's formula: $\hat{x}_\theta = (x_t + (1-\bar{\alpha}_t) s_\theta(x_t, t))/\sqrt{\bar{\alpha}_t}$.

All three parameterisations are equivalent up to a noise-level-dependent rescaling. The choice matters numerically: ε-parameterisation has bounded targets (noise is $\mathcal{N}(0, I)$ regardless of noise level), while x_0-parameterisation has targets of varying scale. EDM (Karras et al. 2022) showed that adaptive scaling of targets and inputs across noise levels is critical for sample quality.

The SDE Perspective: Stochastic vs Deterministic Generation

The same trained score network $s_\theta(x_t, t)$ supports two generation strategies:

Stochastic (DDPM): $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t + \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon_\theta(x_t, t)) + \sqrt{\beta_t}\,\xi, \quad \xi \sim \mathcal{N}(0, I)$

Deterministic (DDIM / probability flow ODE): $x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\,\hat{x}_0(x_t, t) + \sqrt{1-\bar{\alpha}_{t-1}}\,\varepsilon_\theta(x_t, t)$

The deterministic trajectory is the Euler integration of the probability flow ODE. Setting $\eta = 0$ (no noise) in the DDIM update gives the probability flow ODE discretisation. Setting $\eta = 1$ recovers DDPM. Any $\eta \in [0, 1]$ interpolates between the two, trading diversity (high η) for speed and consistency (low η).

The advantage of deterministic generation: the mapping from noise x_T to sample x_0 is *injective*—the same noise always produces the same sample. This enables latent space manipulation: editing x_T produces controlled edits to x_0 (e.g. style transfer, inpainting, interpolation).

Variance Reduction in Diffusion Training

The denoising objective $\mathcal{L} = \mathbb{E}_{t, x_0, \varepsilon}[\lambda(t)\|\varepsilon - \varepsilon_\theta(\cdot)\|^2]$ has high gradient variance due to the randomness in t and ε. Three variance reduction techniques:

Uniform weighting vs SNR weighting: The VLB weights each timestep by $\lambda(t) \propto 1/\text{SNR}(t)$ (signal-to-noise ratio), upweighting high-noise steps. Ho et al. found that uniform weighting $\lambda(t) = 1$ trains better in practice, because the perceptually important low-noise steps are upweighted.

Stratified sampling of t: Rather than sampling $t \sim \text{Uniform}(1, T)$ independently, stratify into B equal-width intervals and sample one t per interval per minibatch. This

ensures all noise levels are represented in every batch, reducing variance in the mean loss over batches by $B\times$.

Loss-weighted sampling (Min-SNR): Hang et al. (2023) propose weighting the loss by $\min(\text{SNR}(t), \gamma)$, clipping the upweighting of high-SNR (low-noise) steps. This "min-SNR-γ" weighting achieves the best of both: training signal from all noise levels, without domination by the high-SNR regime.

Conclusions: The Mathematical Unity of Generative Modelling

The generative paradigm reveals a profound mathematical unity underlying apparently disparate architectures. VAEs, GANs, flows, diffusion models, and consistency models are all solving the same fundamental problem—learning to transport a simple prior distribution to a complex data distribution—through different mathematical mechanisms that trade off computational and statistical properties.

The connections are increasingly well-understood: score matching is the continuous-time limit of denoising autoencoders; diffusion models are continuous normalising flows trained via score matching; flow matching generalises diffusion by replacing the SDE with an ODE; consistency models compress the multi-step ODE into a single-step map; the OT coupling makes this map maximally straight.

The open frontier is the theory of quality: why do diffusion models outperform VAEs for images despite the VAE's more principled probabilistic foundation? The answer involves the interplay between model capacity, training dynamics, and the geometric structure of the data manifold—questions that stochastic interpolants and flow matching are beginning to address rigorously. Future work connecting the Fisher metric of information geometry (Part I) to the curvature of flow trajectories may provide a unified theory of generative model quality.

Appendix: Key Formulae Reference

A.1 VAE Objectives

ELBO: $\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}(q_\phi(z|x) \| p(z))$

Reparametrised gradient: $\nabla_\phi \mathbb{E}_{q_\phi}[f(z)] = \mathbb{E}_\varepsilon[\nabla_\phi f(\mu_\phi + \sigma_\phi \odot \varepsilon)]$

β-VAE: $\mathcal{L}_\beta = \mathbb{E}[\log p_\theta(x|z)] - \beta\, D_{KL}(q_\phi(z|x) \| p(z))$

VQ-VAE: $\mathcal{L} = \|x - D(z_q)\|^2 + \|sg[z_e] - z_q\|^2 + \beta \|z_e - sg[z_q]\|^2$

A.2 GAN Objectives

Original GAN: $\min_G \max_D \mathbb{E}_{x\sim p_r}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))]$

WGAN-GP: $\mathcal{L}_D = \mathbb{E}[f_w(x)] - \mathbb{E}[f_w(G(z))] + \lambda \mathbb{E}[(\|\nabla_{\hat{x}} f_w(\hat{x})\|_2 - 1)^2]$

Optimal discriminator: $D^*_G(x) = p_r(x)/(p_r(x) + p_g(x))$

A.3 Diffusion Model Objectives

DDPM forward marginal: $q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I), \quad \bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$

DDPM simplified objective: $\mathcal{L}_{\text{simple}} = \mathbb{E}_{t,x_0,\varepsilon}[\|\varepsilon - \varepsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\varepsilon, t)\|^2]$

DDIM update: $x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{x}_0(x_t) + \sqrt{1-\bar{\alpha}_{t-1}}\varepsilon_\theta(x_t, t)$

Classifier-free guidance: $\tilde{\varepsilon}_\theta(x_t, t, y) = \varepsilon_\theta(x_t, t, \emptyset) + \gamma(\varepsilon_\theta(x_t, t, y) - \varepsilon_\theta(x_t, t, \emptyset))$

A.4 Flow Matching

CFM objective: $\mathcal{L}_{\text{CFM}} = \mathbb{E}_{t,x_0,x_1}[\|v_\theta((1-t)x_0 + tx_1, t) - (x_1 - x_0)\|^2]$

Probability flow ODE: $dx/dt = f(x,t) - \frac{1}{2}g(t)^2\nabla_x \log p_t(x)$

Continuity equation: $\partial_t p_t + \nabla \cdot (p_t u_t) = 0$

A.5 Self-Supervised Learning

InfoNCE: $\mathcal{L} = -\frac{1}{N}\sum_i \log \frac{e^{\text{sim}(z_i, z_i^+)/\tau}}{\sum_{j \neq i} e^{\text{sim}(z_i, z_j)/\tau}}$

Barlow Twins: $\mathcal{L}_{BT} = \sum_i (1 - C_{ii})^2 + \lambda \sum_{i \neq j} C_{ij}^2$

BYOL: $\mathcal{L}_{\text{BYOL}} = \|q_\theta(z_{\text{online}}) - sg[z_{\text{target}}]\|^2$

MATHEMATICS OF DEEP LEARNING

PART V

REINFORCEMENT LEARNING AND ALIGNMENT

MDP · Bellman · Policy Gradients · MaxEnt RL · Model-Based RL

RLHF · DPO · Constitutional AI · Emergent Properties · Mechanistic Interpretability

Eight-Part Monograph Series on the Mathematical Principles of Neural Networks

Independent of Companion Volumes — Complete in Itself

Silvano Salvador

© Mathematics of Deep Learning Series

PART V

REINFORCEMENT LEARNING AND ALIGNMENT

MDP · Bellman · Policy Gradients · MaxEnt RL · Model-Based RL

RLHF · DPO · Constitutional AI · Emergent Properties · Mechanistic Interpretability

© Mathematics of Deep Learning Series

Silvano Salvador

Introduction: Why Reinforcement Learning Is Fundamentally Different

Historical and Motivational Context. *The earliest formal treatment of sequential decision-making under uncertainty is the work of Bellman (1957) on dynamic programming, which introduced the optimality equations and the principle of backward induction. The connection to machine learning arrived through the temporal difference (TD) methods of Sutton (1988) and the Q-learning algorithm of Watkins (1989). These tabular methods worked on small problems but failed catastrophically when the state space was too large for explicit representation—the "curse of dimensionality" that Bellman himself identified. The deep RL revolution began when Mnih et al. (2015) combined Q-learning with deep convolutional networks and two stabilising tricks (experience replay and target networks), showing superhuman performance on Atari games. The AlphaGo breakthrough (Silver et al., 2016) demonstrated that RL combined with Monte Carlo tree search and self-play could master the game of Go, a space with 10^{170} positions. The post-2017 era extended these methods to robotic control (SAC, TD3), language model fine-tuning (RLHF), and protein structure prediction (AlphaFold2's structure module). What unifies all of these is the Bellman equation and the policy gradient theorem, which are developed in this part from first principles.*

Reinforcement learning addresses the problem of sequential decision-making under uncertainty: an agent interacts with an environment, receives scalar rewards, and must learn a policy that maximises cumulative return. Unlike supervised learning and generative modelling, RL must deal with the fundamental challenge that actions affect future observations, creating temporal credit assignment problems and non-stationarity.

Cross-Reference to Other Parts. The mathematical foundations of RL connect to the information geometry of Part I: the natural gradient policy optimisation methods (TRPO, ACKTR) use the Fisher information metric to define the geometry of the policy space, and the natural policy gradient is the steepest ascent direction in this geometry. The PAC learning theory of Part III provides sample complexity bounds for RL under the concept of PAC-MDP: conditions under which a polynomial number of interactions suffices to learn an ε-optimal policy with high probability.

Markov Decision Processes and the Bellman Equations

Historical and Motivational Context. *The Markov property—the future is conditionally independent of the past given the present—was formalised by Markov (1906) for stochastic processes over countable state spaces. Its application to decision problems yielded the MDP framework (Bellman, 1957; Howard, 1960), which became the canonical model for operations research and later AI planning. The key insight is that the Markov property enables a* recursive *decomposition of the value function: the optimal value of being in state s equals the immediate reward plus the discounted optimal value of the next state, regardless of how one arrived at s. This recursion is the Bellman equation, and it is the foundation of every RL algorithm.*

Formal Specification and the Markov Property

A finite MDP is a tuple (S, A, P, R, γ) where S is a finite state space, A a finite action space, $P : S \times A \times S \to [0,1]$ the transition kernel, $R : S \times A \to \mathbb{R}$ the reward function, and $\gamma \in [0,1)$ the discount factor. A policy $\pi : S \to \Delta(A)$ maps states to distributions over actions. The Markov property states that $P(s_{t+1}|s_0, a_0, \dots, s_t, a_t) = P(s_{t+1}|s_t, a_t)$, making the transition kernel a sufficient statistic of the history.

The discount factor γ has two interpretations: (1) geometric probability of episode termination at each step, giving finite-horizon semantics to an infinite-horizon problem; (2) time preference, expressing that near rewards are worth more than distant rewards. The condition $\gamma < 1$ ensures that the infinite sum $\sum_t \gamma^t r_t$ converges whenever rewards are bounded.

The Bellman Equations: Derivation and Fixed-Point Analysis

The state value function $V^\pi(s) = \mathbb{E}_\pi[\sum_t \gamma^t r_t | s_0 = s]$ and action-value function $Q^\pi(s,a) = \mathbb{E}_\pi[\sum_t \gamma^t r_t | s_0 = s, a_0 = a]$ satisfy the Bellman consistency equations, derived as follows:

$$\begin{aligned} V^\pi(s) &= \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r_t \,\middle|\, s_0 = s\right] = \mathbb{E}_{a\sim\pi(\cdot|s)}\Big[R(s,a) + \gamma\, \mathbb{E}_{s'\sim P(\cdot|s,a)}[V^\pi(s')]\Big] \\ &= \sum_a \pi(a|s) \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a)\, V^\pi(s')\right] \end{aligned} \tag{1}$$

where the second equality uses the tower property of conditional expectations and the stationarity of the Markov chain.

Theorem 1.1 (Bellman Optimality and Contraction)**.** *The optimal value function $V^*(s) = \max_\pi V^\pi(s)$ satisfies the Bellman optimality equation: $V^*(s) = \max_a [R(s,a) + \gamma \sum_{s'} P(s'|s,a)\, V^*(s')]$. The Bellman optimality operator T^*, defined by $(T^*V)(s) = \max_a [R(s,a) + \gamma \sum_{s'} P(s'|s,a)\, V(s')]$, is a γ-contraction in the ℓ^∞ norm: $\|T^*V - T^*U\|_\infty \le \gamma \|V - U\|_\infty$. By the Banach fixed-point theorem, T^* has a unique fixed point V^*, and value iteration $V_{k+1} = T^* V_k$ converges geometrically: $\|V_k - V^*\|_\infty \le \gamma^k \|V_0 - V^*\|_\infty / (1-\gamma)$.*

Proof of the contraction. $|(T^*V)(s) - (T^*U)(s)| = |\max_a [\cdot + \gamma \sum_{s'} PV(s')] - \max_a [\cdot + \gamma \sum_{s'} PU(s')]| \le \gamma \max_a \sum_{s'} P(s'|s,a)|V(s') - U(s')| \le \gamma \|V - U\|_\infty$. Taking the max

over s gives the result. The inequality $|\max f - \max g| \le \max|f-g|$ is used in the first step. □

Q-Learning, TD Learning, and the Deadly Triad

Q-learning (Watkins and Dayan, 1992) is an off-policy TD algorithm:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\big[r_t + \gamma \max_{a'} Q(s_{t+1},a') - Q(s_t,a_t)\big] \tag{2}$$

Convergence to Q^* is guaranteed for tabular representations under mild conditions (every (s,a) visited infinitely often, diminishing learning rate). With function approximation, however, Q-learning can diverge—this is part of the *deadly triad* (Sutton and Barto, 2018): the combination of (1) off-policy learning, (2) function approximation, and (3) bootstrapping (using the current estimate as a target) can cause instability.

DQN (Mnih et al., 2015) stabilises deep Q-learning with:

- **Experience replay**: store transitions (s,a,r,s') in a buffer $\mathcal{D}$; sample random mini-batches to break temporal correlations.
- **Target networks**: maintain a slowly-updated copy Q_{θ^-} to stabilise the target $y = r + \gamma \max_{a'} Q_{\theta^-}(s',a')$, preventing the non-stationarity of self-referential updates.

Double DQN decouples action selection from evaluation: $y = r + \gamma\, Q_{\theta^-}(s', \arg\max_{a'} Q_\theta(s',a'))$, reducing overestimation bias that arises because max of noisy estimates exceeds the estimate of the max.

Distributional RL and Risk-Sensitive Criteria

Standard RL maximises expected return. Distributional RL (Bellemare et al., 2017) maintains the full return distribution $Z^\pi(s,a) \overset{d}{=} \sum_t \gamma^t r_t$. The distributional Bellman operator is: $Z^\pi(s,a) \overset{d}{=} R(s,a) + \gamma\, Z^\pi(s', \pi(s'))$ where $\overset{d}{=}$ denotes equality in distribution. C51 parameterises the return distribution as a categorical distribution over a fixed support $\{v_1, \dots, v_{51}\}$. The Wasserstein contraction property $W_p(\mathcal{T}^\pi Z, \mathcal{T}^\pi Z') \le \gamma\, W_p(Z,Z')$ guarantees convergence.

The full distribution enables risk-sensitive policies: a CVaR-constrained policy $\pi^*_\alpha = \arg\max_\pi \mathrm{CVaR}_\alpha(Z^\pi)$ avoids tail losses that a mean-maximising policy would accept.

> **Cross-Reference to Other Parts.** The Wasserstein contraction used here is the same metric studied in Part IV for GANs and flow matching. The distributional perspective on returns is the RL analogue of the score-based generative model perspective on data: rather than estimating a scalar, estimate a distribution. The connection to optimal transport (Part IV, Chapter 7) is explicit: the distributional Bellman operator is a contraction in the W_p metric, which is the metric that flow matching minimises in the generative setting.

Policy Gradients

Historical and Motivational Context. *Policy gradient methods originate with REINFORCE (Williams, 1992), which derived the score function gradient estimator for the expected return. The fundamental problem—high variance of the gradient estimate—plagued early methods and led to the development of actor-critic architectures (Sutton et al., 1999), which combine a policy (actor) with a value function estimate (critic) to reduce variance via the advantage function. The theoretical breakthrough was the Policy Gradient Theorem (Sutton et al., 2000), which showed that the gradient of the expected return with respect to policy parameters takes a clean form independent of the transition dynamics. TRPO (Schulman et al., 2015) addressed the step size problem by constraining updates to a KL ball; PPO (Schulman et al., 2017) achieved the same stability with a simpler clipped objective that became the workhorse of language model fine-tuning.*

The Policy Gradient Theorem: Full Derivation

Theorem 2.1 (Policy Gradient Theorem, Sutton et al. 2000)**.** *Let $J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[\sum_t \gamma^t r_t]$ be the expected discounted return. Then:*

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_t \nabla_\theta \log \pi_\theta(a_t|s_t)\cdot Q^{\pi_\theta}(s_t,a_t)\right] \tag{3}$$

This holds for any differentiable policy and any MDP, regardless of the transition dynamics P. The gradient does not involve $\nabla_\theta P$.

Derivation. $J(\theta) = \sum_s d^\pi(s)\sum_a \pi_\theta(a|s)Q^\pi(s,a)$ where $d^\pi(s) = (1-\gamma)\sum_{t=0}^\infty \gamma^t P(s_t = s|\pi)$ is the discounted state visitation measure. Taking the gradient:

$$\begin{aligned}
\nabla_\theta J(\theta) &= \sum_s d^\pi(s)\sum_a \nabla_\theta \pi_\theta(a|s)\, Q^\pi(s,a) \\
&= \sum_s d^\pi(s)\sum_a \pi_\theta(a|s)\,\nabla_\theta \log \pi_\theta(a|s)\, Q^\pi(s,a) \\
&= \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(a|s)\, Q^\pi(s,a)]
\end{aligned} \tag{4}$$

The key step is writing $\nabla_\theta \pi_\theta = \pi_\theta \cdot \nabla_\theta \log \pi_\theta$ (the log-derivative trick). The transition dynamics P appear only in d^π, but since $\nabla_\theta d^\pi(s)$ cancels in the complete derivation (see Sutton et al. for details via the policy gradient identity $\sum_{s'} \nabla_\theta d^\pi(s') = 0$ in the undiscounted formulation), the gradient is independent of the environment model. □

The *baseline subtraction identity*: for any function $b(s)$ depending only on state, $\mathbb{E}[\nabla_\theta \log \pi_\theta(a|s)\cdot b(s)] = 0$. *Proof*: $\sum_a \nabla_\theta \pi_\theta(a|s)\cdot b(s) = b(s)\nabla_\theta \sum_a \pi_\theta(a|s) = b(s)\nabla_\theta 1 = 0$. This allows subtracting any state-dependent baseline: $\nabla_\theta J = \mathbb{E}[\nabla_\theta \log \pi_\theta(a_t|s_t)\cdot A^\pi(s_t,a_t)]$ where $A^\pi = Q^\pi - V^\pi$ is the advantage function. The advantage has lower variance than Q because it measures the relative quality of an action, not its absolute return.

PPO: Proximal Policy Optimisation — Full Derivation

TRPO (Schulman et al., 2015) maximises the surrogate objective $\hat{L}(\theta) = \mathbb{E}_t[r_t(\theta)\hat{A}_t]$ subject to $\mathbb{E}_t[D_{KL}(\pi_{\theta_{\text{old}}}\|\pi_\theta)] \le \delta$, where $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{\text{old}}}(a_t|s_t)$ is the importance ratio. Solving this constrained problem requires the conjugate gradient method to compute the natural gradient and a line search, making it computationally expensive.

PPO replaces the KL constraint with a clipped objective:

$$\mathcal{L}_{\text{CLIP}}(\theta) = \mathbb{E}_t\Big[\min\Big(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t\Big)\Big] \tag{5}$$

Case analysis. When $\hat{A}_t > 0$ (action was better than expected):

- $r_t < 1-\varepsilon$: the action is underweighted; the objective rewards increasing r_t (no clip active).
- $1-\varepsilon \le r_t \le 1+\varepsilon$: the objective rewards increasing r_t normally.
- $r_t > 1+\varepsilon$: the objective is clipped at $(1+\varepsilon)\hat{A}_t$; no gradient incentive to increase r_t further. This prevents the policy from moving too far.

When $\hat{A}_t < 0$ (action was worse than expected), the analysis is symmetric: clipping at $1-\varepsilon$ prevents excessive downweighting of the action.

The complete PPO objective adds a value function loss and entropy bonus:

$$\mathcal{L}^{\text{PPO}}(\theta) = \mathcal{L}_{\text{CLIP}}(\theta) - c_1\mathcal{L}_{VF}(\theta) + c_2 H[\pi_\theta](\cdot|s_t) \tag{6}$$

where c_1, c_2 are coefficients, $\mathcal{L}_{VF} = (V_\theta(s_t) - V_t^{\text{targ}})^2$ is the value loss, and H is the entropy bonus that encourages exploration.

The GAE (Generalised Advantage Estimation, Schulman et al., 2016) computes the advantage as a λ-weighted sum of n-step TD errors:

$$\hat{A}_t = \sum_{l=0}^{\infty} (\gamma\lambda)^l\, \delta_{t+l}^V, \quad \delta_t^V = r_t + \gamma V(s_{t+1}) - V(s_t) \tag{7}$$

$\lambda = 0$ gives one-step TD (low variance, high bias); $\lambda = 1$ gives Monte Carlo returns (high variance, no bias). The exponential weighting provides a smooth bias-variance trade-off.

Cross-Reference to Other Parts. The PPO clipped objective is a first-order approximation to the TRPO trust region constraint. The connection to information geometry (Part I) is explicit: the KL divergence between old and new policies measures the "distance" in the space of distributions, and the natural gradient direction (Fisher information metric) is precisely the direction that maintains a fixed KL ball. ACKTR (Wu et al., 2017) applies the Kronecker-factored approximation of Part I (K-FAC) to the Fisher matrix of the policy, achieving TRPO-level stability with PPO-level simplicity.

Maximum Entropy RL: SAC and TD3

Historical and Motivational Context. *The entropy-regularised perspective on RL has roots in the Boltzmann distribution over policies (Todorov, 2006) and the connection between maximum entropy inference and optimal control (Ziebart et al., 2008). The MaxEnt framework gained widespread adoption after Haarnoja et al. (2018) showed that the Soft Actor-Critic algorithm—which operationalises MaxEnt RL for continuous actions—achieved state-of-the-art sample efficiency on MuJoCo benchmarks with a fraction of the environment interactions required by PPO.*

The key theoretical insight is that the entropy bonus induces a Boltzmann distribution over actions, which automatically handles multi-modal optimal policies and provides a principled exploration mechanism.

The MaxEnt Objective and Soft Bellman Equations

The MaxEnt RL objective is: $J(\pi) = \sum_t \mathbb{E}[r(s_t, a_t) + \alpha\, H(\pi(\cdot|s_t))]$ where $H(\pi(\cdot|s)) = -\sum_a \pi(a|s) \log \pi(a|s)$ is the policy entropy and α is the temperature parameter.

Theorem 3.1 (Optimal MaxEnt Policy)**.** *The policy maximising the MaxEnt objective with temperature α is: $\pi^*(a|s) \propto \exp(Q^*_{soft}(s,a)/\alpha)$. This is a Gibbs distribution over actions, with temperature controlling exploitation–exploration. The optimal policy satisfies a soft policy improvement theorem: for any policy π, the policy $\tilde{\pi}(a|s) \propto \exp(Q^{\pi}(s,a)/\alpha)$ achieves $V^{\tilde{\pi}}_{soft}(s) \geq V^{\pi}_{soft}(s)$ everywhere.*

Derivation. The soft Bellman equations are:

$$\begin{aligned} V^{\pi}_{\text{soft}}(s) &= \mathbb{E}_{a\sim\pi}[Q^{\pi}_{\text{soft}}(s,a)] + \alpha\, H(\pi(\cdot|s)) \\ Q^{\pi}_{\text{soft}}(s,a) &= R(s,a) + \gamma\, \mathbb{E}_{s'\sim P}[V^{\pi}_{\text{soft}}(s')] \end{aligned} \tag{8}$$

Setting $\nabla_\pi[V^{\pi}_{\text{soft}}] = 0$ with the constraint $\sum_a \pi(a|s) = 1$ via Lagrange multiplier λ: $\nabla_\pi[\mathbb{E}[Q^{\pi}_{\text{soft}} - \alpha \log \pi] + \lambda(\sum_a \pi - 1)] = 0 \Rightarrow Q^{\pi}_{\text{soft}}(s,a) - \alpha(\log \pi(a|s) + 1) + \lambda = 0$ $\Rightarrow \log \pi^*(a|s) = Q^{\pi}_{\text{soft}}(s,a)/\alpha + \text{const} \Rightarrow \pi^*(a|s) = \exp(Q^{\pi}_{\text{soft}}(s,a)/\alpha)/Z(s)$. □

Soft Actor-Critic (SAC)

SAC (Haarnoja et al., 2018) implements MaxEnt RL for continuous actions with a squashed Gaussian policy: $a_t = \tanh(\mu_\theta(s_t) + \sigma_\theta(s_t) \odot \varepsilon)$, $\varepsilon \sim \mathcal{N}(0, I)$. The log-probability of a under this policy requires the change-of-variables formula for the tanh squashing: $\log \pi_\theta(a|s) = \log \mathcal{N}(\text{arctanh}(a); \mu, \sigma^2) - \sum_i \log(1 - a_i^2)$.

The actor maximises: $J_\pi(\theta) = \mathbb{E}_{s\sim\mathcal{D},\, \varepsilon\sim\mathcal{N}}[\alpha \log \pi_\theta(f_\theta(\varepsilon; s)|s) - \min(Q_{w_1}(s, f_\theta(\varepsilon; s)), Q_{w_2}(s, f_\theta(\varepsilon; s)))]$

The temperature α is tuned automatically by minimising: $J(\alpha) = \mathbb{E}[-\alpha \log \pi_\theta(a_t|s_t) - \alpha \bar{H}]$ where $\bar{H}$ is a target entropy (typically $-\dim(\mathcal{A})$).

Model-Based RL: World Models and Latent Planning

Historical and Motivational Context. *The dream of model-based RL—learning a model and planning in it—is as old as RL itself: Sutton (1990) introduced Dyna, which interleaves model-free updates with simulated experience generated from a learned model. The fundamental obstacle is model exploitation: the policy learns to exploit errors in the model, achieving high reward in the model while failing in the real environment. Early model-based methods (linear models, Gaussian processes) were limited to low-dimensional spaces. The latent-space approach (World Models, Ha and Schmidhuber, 2018; Dreamer, Hafner et al., 2020) overcame this by learning a compact latent representation in which both the world model and the policy operate.*

The MBPO Error Bound and Short Rollout Principle

Theorem 4.1 (MBPO Branched Rollout Bound, Janner et al. 2019)**.** *Let* $\varepsilon_m = \max_t \mathbb{E}_{s\sim\pi}[D_{TV}(p(s'|s,a)\|\hat{p}$ *be the model error and* ε_π *the policy error. For* k*-step model rollouts branching from real data, the return difference is bounded by:*

$$|\eta[\pi] - \eta_{model}[\pi]| \leq \frac{2\gamma(\gamma^k\varepsilon_m + (1-\gamma^k)(\varepsilon_m + \varepsilon_\pi))}{(1-\gamma)^2} \tag{9}$$

The bound is minimised at a short rollout length $k = O(\log(\varepsilon_m)/\log(\gamma))$*, independent of policy complexity.*

This justifies MBPO's key design: short (1–5 step) model rollouts that generate synthetic transitions for off-policy RL, gaining sample efficiency without catastrophic model exploitation.

Dreamer and the RSSM

DreamerV3 (Hafner et al., 2023) learns a world model with a Recurrent State Space Model (RSSM) factoring the latent state into deterministic $h_t = f(h_{t-1}, z_{t-1}, a_{t-1})$ and stochastic $z_t \sim q_\phi(z_t|h_t, o_t)$ components. The world model is trained by maximising the ELBO of the observation sequence:

$$\mathcal{L}_{\text{model}} = \mathbb{E}_q\left[\sum_t \log p(o_t|h_t, z_t) - \beta\, D_{KL}(q(z_t|h_t, o_t)\|p(z_t|h_t))\right] \tag{10}$$

Policies and value functions are trained entirely in imagination. DreamerV3 introduces symlog predictions (symmetric log transform applied to targets) and free bits (KL clipped from below), enabling training with a single set of hyperparameters across tasks spanning orders of magnitude in reward scale.

Cross-Reference to Other Parts. The RSSM is a structured state-space model whose inference network is a variational autoencoder (Part IV, Chapter 1). The ELBO for the observation sequence is the sequential analogue of the image ELBO, with the KL term penalising the divergence between the posterior (conditioned on observations) and the prior (conditioned only on actions and history). The connection to score-based generation (Part IV, Chapter 4) is that the transition prior $p(z_t|h_t)$ can be replaced by a diffusion prior for improved sample quality.

RLHF, DPO, and the Mathematics of Alignment

Historical and Motivational Context. *The alignment problem—ensuring that learned systems pursue human-intended objectives—was identified as a central challenge by Norbert Wiener (1960) in his reflection on the "sorcerer's apprentice" problem: a system that optimises an objective proxy rather than the true objective may find solutions that satisfy the proxy in unintended ways. The formal treatment via inverse reward design (Hadfield-Menell et al., 2016) and cooperative inverse reinforcement learning (Hadfield-Menell et al., 2016) provided game-theoretic frameworks. RLHF (Christiano et al., 2017) operationalised alignment for neural networks by replacing the unknown reward with a learned reward model trained on human preference comparisons. The appli-*

cation of RLHF to large language models (Ziegler et al., 2019; InstructGPT, Ouyang et al., 2022; ChatGPT, 2022) demonstrated that the method could substantially improve model helpfulness and harmlessness—but required expensive human annotation and a complex three-stage pipeline. DPO (Rafailov et al., 2023) simplified this pipeline dramatically, eliminating the reward model entirely.

The Bradley-Terry Model and Reward Learning

RLHF assumes that human preferences between response pairs (y_w, y_l) given prompt x follow the Bradley-Terry model:

$$p^*(y_w \succ y_l | x) = \sigma(r^*(x, y_w) - r^*(x, y_l)) \tag{11}$$

where $r^*(x, y)$ is the latent reward function and σ is the logistic function. A reward model $r_\phi(x, y)$ is trained by maximum likelihood:

$$\mathcal{L}_{RM}(\phi) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}[\log \sigma(r_\phi(x, y_w) - r_\phi(x, y_l))] \tag{12}$$

The trained reward model is used to fine-tune the policy via PPO, maximising: $\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}[r_\phi(x, y)] - \beta\, D_{KL}(\pi_\theta \| \pi_{\text{ref}})$ where π_{ref} is the reference (supervised fine-tuned) policy.

Theorem 5.1 (Optimal RLHF Policy)**.** *The optimal policy for the KL-constrained objective* $\max_\pi \mathbb{E}[r(x, y)] - \beta\, D_{KL}(\pi \| \pi_{ref})$ *has the closed-form solution:*

$$\pi^*(y|x) = \frac{\pi_{ref}(y|x)\exp(r(x, y)/\beta)}{Z(x)} \tag{13}$$

where $Z(x) = \sum_y \pi_{ref}(y|x)\exp(r(x, y)/\beta)$ *is the partition function.*

Derivation. The objective can be written as: $\mathcal{F}(\pi) = \mathbb{E}_{y\sim\pi}[r(x, y)] - \beta\, D_{KL}(\pi \| \pi_{\text{ref}}) = -\beta\, D_{KL}(\pi \| \tilde{\pi}) + \text{const}$ where $\tilde{\pi}(y|x) \propto \pi_{\text{ref}}(y|x)\exp(r(x, y)/\beta)$. The minimum KL is achieved when $\pi = \tilde{\pi}$. □

Direct Preference Optimisation: Complete Derivation from Bradley-Terry

DPO (Rafailov et al., 2023) eliminates the reward model by expressing the reward in terms of policy ratios. From Theorem 5.1:

$$r(x, y) = \beta \log \frac{\pi^*(y|x)}{\pi_{\text{ref}}(y|x)} + \beta \log Z(x) \tag{14}$$

Substituting into the Bradley-Terry preference probability:

$$\begin{aligned} p^*(y_w \succ y_l | x) &= \sigma(r^*(x, y_w) - r^*(x, y_l)) \\ &= \sigma\left(\beta \log \frac{\pi^*(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi^*(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right) \end{aligned} \tag{15}$$

where the $\beta \log Z(x)$ terms cancel because $Z(x)$ depends only on x. Replacing π^* with the parameterised policy π_θ, the DPO objective (maximum likelihood under the Bradley-Terry model) is:

$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)\right] \tag{16}$$

This is a binary cross-entropy objective over implicit reward differences, requiring no explicit reward model, no RL loop, and no sampling during training. The implicit reward of DPO is $\hat{r}_\theta(x, y) = \beta \log(\pi_\theta(y|x)/\pi_{\text{ref}}(y|x))$.

Gradient analysis. The DPO gradient is:

$$\nabla_\theta \mathcal{L}_{\text{DPO}} = -\beta\, \mathbb{E}[\sigma(\hat{r}_\theta(y_l) - \hat{r}_\theta(y_w))\, (\nabla_\theta \log \pi_\theta(y_w|x) - \nabla_\theta \log \pi_\theta(y_l|x))] \tag{17}$$

The weight $\sigma(\hat{r}_\theta(y_l) - \hat{r}_\theta(y_w))$ is high when the model incorrectly rates y_l over y_w, focusing gradient updates on the most misranked pairs.

KTO, SimPO, and Extensions

KTO (Ethayarajh et al., 2024) replaces pairwise preferences with a prospect-theoretic framework. Humans weight losses more heavily than gains (loss aversion), so the utility function has an asymmetric shape:

$$\mathcal{L}_{\text{KTO}}(\theta) = \mathbb{E}[(\lambda_D - \sigma(\hat{r}(x, y) - z_0)) \cdot \mathbf{1}[y \text{ desirable}] + (\lambda_U - \sigma(z_0 - \hat{r}(x, y))) \cdot \mathbf{1}[y \text{ undesirable}]] \tag{18}$$

where $z_0 = \mathbb{E}_{x' \sim \mathcal{D}}[D_{KL}(\pi_\theta(\cdot|x') \| \pi_{\text{ref}}(\cdot|x'))]$ is the KL reference point. KTO requires only binary quality labels, reducing annotation cost substantially.

SimPO (Meng et al., 2024) uses length-normalised rewards and a target margin γ, eliminating the reference model: the objective is $-\log \sigma((\beta/|y_w|) \log \pi_\theta(y_w|x) - (\beta/|y_l|) \log \pi_\theta(y_l|x) - \gamma)$.

Constitutional AI and Scalable Oversight

Constitutional AI (Bai et al., 2022) addresses the scalability of human oversight. The approach has two phases:

1. **SL-CAI**: The model critiques its own outputs according to a constitution of principles and revises accordingly. This is a self-improvement loop that converges when the model's outputs satisfy the principles in expectation.
2. **RLAIF**: An AI classifier replaces human annotators for preference labelling, scaling oversight to more data than human annotation could provide.

The mathematical structure is a fixed-point iteration in policy space: $\pi_{k+1} = \text{SFT}(\mathcal{C}(\pi_k))$ where $\mathcal{C}$ is the critique-revision operator. Convergence to a fixed point is guaranteed if $\mathcal{C}$ is a contraction—a condition that holds when the constitution provides a well-defined total order over response quality.

Cross-Reference to Other Parts. The RLHF pipeline—supervised fine-tuning, reward modelling, RL fine-tuning—is a three-stage learning algorithm that combines all three learning paradigms: supervised learning (SFT), learning from pairwise comparisons (reward model), and RL with function approximation (PPO). The generalisation theory of Part III applies to each stage: the SFT model generalises by the classical VC bounds; the reward model generalises via the learning-to-rank theory; the PPO fine-tuning is bounded by the MBPO-style return difference bound of Chapter 4.

Mathematics of Deep Learning — Part I: The Geometric Foundations

Emergent Properties and In-Context Learning

Historical and Motivational Context. *The term "emergent" in the context of neural network capabilities was formalised by Wei et al. (2022), who documented abilities that are absent in smaller models and appear suddenly at larger scales—phase transitions in capability. The most striking examples include multi-step arithmetic (absent below 13B parameters, near-perfect above), multi-language translation, and chain-of-thought reasoning. The theoretical explanation for why these transitions are sharp (rather than gradual) remains an open question, though sharp transitions have been observed in statistical mechanics (the Ising model phase transition, the storage capacity transition in Hopfield networks) and are associated with percolation phenomena in random graphs.*

In-Context Learning as Implicit Bayesian Inference

ICL presents a model with k examples $(x_1, y_1), \dots, (x_k, y_k)$ and a query x_{k+1}, predicting y_{k+1} without parameter updates.

Xie et al. (2022) model ICL as Bayesian inference: the pretraining distribution is a mixture $p(x) = \sum_C p(C)p(x|C)$ over latent concepts, and the language model implicitly maintains a posterior $p(C|x_1, \dots, x_k)$ that concentrates on the true concept as examples arrive. The ICL prediction is: $p_{LM}(y|\text{context}) \approx \mathbb{E}_{C \sim p(C|\text{context})}[p(y|x_{k+1}, C)]$.

Akyürek et al. (2023) provide a mechanistic interpretation: transformer layers can implement gradient descent steps, so ICL corresponds to running ridge regression in the forward pass. For a specific one-hidden-layer transformer, the attention update $W_Q^T W_K$ implements a gradient descent step on the in-context examples, with W_V learning the feature-to-label map.

Chain-of-Thought and Computational Depth

Standard transformers with constant depth and width are limited to the circuit complexity class TC^0 (constant-depth threshold circuits). Feng et al. (2023) show that chain-of-thought reasoning allows the network to compute functions outside TC^0—specifically up to NC^1 (log-depth circuits). The intermediate tokens function as a scratchpad increasing effective computational depth, enabling tasks (counting, sorting, arithmetic) that require depth super-constant in input length.

Cross-Reference to Other Parts. The phase transition perspective on emergent abilities connects to the statistical physics analysis of Part VI (Chapter 1): the capacity transition in Hopfield networks (from the retrieval phase to the spin glass phase) is a sharp first-order transition with the same mathematical structure as the capability emergence in large language models. The Amari–Nagaoka information geometry of Part I provides a differential-geometric interpretation of the "sharpness" of phase transitions in terms of the curvature of the log-likelihood landscape.

Mechanistic Interpretability

Historical and Motivational Context. *Mechanistic interpretability grew out of network dissection (Bau et al., 2017) and feature visualisation (Olah et al., 2017), which demonstrated that individual neurons respond to interpretable concepts. The circuits programme (Olah et al., 2020) deepened this to* circuits*: small computational subgraphs implementing recognisable functions. The discovery of induction heads (Olsson et al., 2022) provided a concrete mechanistic explanation for in-context learning in terms of a two-layer attention circuit. Sparse autoencoders (Cunningham et al., 2023; Bricken et al., 2023) addressed the polysemanticity problem—individual neurons typically respond to multiple unrelated concepts—by learning a sparse dictionary that decomposes activations into monosemantic features.*

Circuits and the Modularity Hypothesis

The circuits hypothesis (Olah et al., 2020) proposes that neural networks implement identifiable, modular computational circuits. A circuit is formally a subgraph (N, E) of the full computation graph $G = (V, E)$ sufficient to explain a specific behaviour, in the sense that ablating complement nodes does not alter the behaviour on the relevant distribution. Key discovered circuits include: curve detectors (V1-like cells in CNNs), multimodal neurons that respond to the same concept across different modalities, and the indirect object identification circuit (a 26-head circuit implementing grammatical role binding in GPT-2).

Superposition and the Geometry of Features

The superposition hypothesis (Elhage et al., 2022) proposes that networks represent more features than they have dimensions by encoding features as near-orthogonal directions. If there are F features but only $d \ll F$ dimensions, the network stores $O(F)$ features at interference rate $O(F/d)$ per feature.

Theorem 7.1 (Superposition Capacity)**.** *For sparse features (each present with probability $p \ll 1$), the total interference loss is: $\mathbb{E}[\text{loss}] = \sum_i (1 - \|u_i\|^2)^2 I_i + \sum_{i \neq j} (u_i \cdot u_j)^2 I_i I_j p$ When $d < F$, the minimum is achieved by a polytope code: features placed at vertices of a regular polytope in $\mathbb{R}^d$. For $p \to 0$, interference vanishes and $F/d \to \infty$, enabling unbounded superposition.*

Sparse Autoencoders for Feature Discovery

Sparse autoencoders (Cunningham et al., 2023) learn a dictionary of $M \gg d$ feature directions:

$$\mathcal{L}_{\text{SAE}} = \|h - D \cdot \text{ReLU}(Eh + b)\|^2 + \lambda \|\text{ReLU}(Eh + b)\|_1 \tag{19}$$

where $E \in \mathbb{R}^{M \times d}$ and $D \in \mathbb{R}^{d \times M}$ (unit-norm columns). The L^1 penalty induces sparsity, decomposing the polysemantic activation vector h into a sparse sum of monosemantic feature vectors. Features discovered by SAEs are human-interpretable at a granularity not achievable by examining individual neurons.

Induction Heads and the Mechanistic Basis of ICL

Induction heads (Olsson et al., 2022) implement in-context copying via a two-layer circuit: (1) a *previous token head* in layer 0 that attends to the token immediately preceding

the current position, writing the previous token into the residual stream; (2) an *induction head* in layer 1 that uses Q-K matching to find where the current token appeared earlier in context, attending to the position just after that earlier match.

The circuit implements the pattern: given $\ldots A \ldots AB$, predict B. This two-layer circuit is sufficient for arbitrary in-context sequence completion and represents the mechanistic basis for ICL. The circuit's existence can be verified by causal intervention: patching the residual stream at the previous token head's output into a clean run predicts the correct completion with near-zero error.

Cross-Reference to Other Parts. Induction heads are a concrete instance of the associative memory framework developed for Modern Hopfield Networks in Part IV (Chapter 9): the key–query matching in the induction head implements one step of the Hopfield update, with exponential storage capacity in the sequence length. The SAE decomposition of activations connects to the information-geometric notion of sufficient statistics from Part I: the SAE features are approximately minimal sufficient statistics for the input, analogous to the natural parameters of an exponential family.

GRPO, REINFORCE Leave-One-Out, and Recent Alignment Methods

Historical and Motivational Context. *The cost of RLHF with PPO is substantial: it requires four models in memory simultaneously (the policy, the reference policy, the reward model, and the value function), making it memory-intensive and slow. DeepSeek-R1 (DeepSeek-AI, 2025) demonstrated that a simpler algorithm—Group Relative Policy Optimisation (GRPO, Shao et al., 2024)—could achieve comparable or superior reasoning performance by eliminating the value function entirely, using the group mean as a baseline instead.*

Group Relative Policy Optimisation (GRPO)

GRPO (Shao et al., 2024) eliminates the value function by using the group mean as a baseline. For each prompt q, sample G responses $\{o_1, \ldots, o_G\}$ from the current policy π_θ. The advantage of response o_i: $\hat{A}_i = (r_i - \text{mean}(\{r_j\}))/\,\text{std}(\{r_j\})$. The GRPO objective:

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\mathbb{E}\left[\frac{1}{G}\sum_i \sum_t \min(r_{i,t}\hat{A}_i, \text{clip}(r_{i,t}, 1-\varepsilon, 1+\varepsilon)\hat{A}_i)\right] + \beta D_{KL}(\pi_\theta \| \pi_{\text{ref}}) \quad (20)$$

where $r_{i,t} = \pi_\theta(o_{i,t}|q, o_{i,<t})/\pi_{\theta_{\text{old}}}(o_{i,t}|q, o_{i,<t})$. Key advantage: no value network, reducing memory from 4 models to 3.

Process Reward Models and Step-Level Feedback

Process Reward Models (PRMs, Lightman et al., 2023) assign rewards to individual reasoning steps, providing denser supervision. The PRM objective: $\mathcal{L}_{\text{PRM}}(\phi) = -\sum_i \log \sigma(r_\phi(q, s_{1:i}) \cdot l_i)$ where $l_i \in \{+1, -1\}$ are step-level correctness labels. Best-of-N with PRM scoring at inference time—select the response with highest PRM score from N samples—outperforms majority voting by distinguishing correct-reasoning-wrong-answer from incorrect-reasoning-correct-answer.

GRPO, REINFORCE Leave-One-Out, and Recent Alignment Methods: Derivations

GRPO (Shao et al., 2024) eliminates the value function by using the group mean as a baseline. For each prompt q, sample G responses $\{o_1, \ldots, o_G\}$ from the current policy π_θ. The advantage of response o_i is estimated as its standardised reward:

$$\hat{A}_i = \frac{r_i - \text{mean}(\{r_1, \ldots, r_G\})}{\text{std}(\{r_1, \ldots, r_G\})} \tag{21}$$

The GRPO objective is:

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\mathbb{E}_{q,\{o_i\}}\left[\frac{1}{G}\sum_i \sum_t \min(r_{i,t}\hat{A}_i, \text{clip}(r_{i,t}, 1-\varepsilon, 1+\varepsilon)\hat{A}_i)\right] + \beta D_{KL}(\pi_\theta \| \pi_{\text{ref}}) \tag{22}$$

where $r_{i,t} = \pi_\theta(o_{i,t}|q, o_{i,<t}) / \pi_{\theta_{\text{old}}}(o_{i,t}|q, o_{i,<t})$ is the per-token importance ratio.

Key differences from PPO.

- **No value function:** The group mean replaces $V(s)$ as the baseline. This eliminates the value network and its associated memory cost.
- **Group-based advantage:** The advantage measures how much better response o_i is relative to the other $G-1$ responses—a relative rather than absolute quality measure.
- **Group sampling:** Multiple responses per prompt expose the model to diverse outcomes from the same starting point, providing richer gradient signal.

REINFORCE Leave-One-Out (RLOO)

RLOO (Kool et al., 2019; Ahmadian et al., 2024 for LLMs) is another variance-reduction technique for REINFORCE that uses $G-1$ other samples from the same prompt as the baseline:

$$\hat{A}_i^{\text{RLOO}} = r_i - \frac{1}{G-1}\sum_{j \neq i} r_j \tag{23}$$

Unlike the group standard deviation in GRPO, RLOO uses the raw mean of other samples, which is an unbiased estimate of the value function $V^\pi(q)$ (since each r_j is an unbiased sample of the return from state q).

Comparison. GRPO standardises by the group std, while RLOO subtracts the leave-one-out mean. RLOO has provably unbiased advantage estimates; GRPO's normalisation is a heuristic that controls gradient scale. In practice, both perform similarly, with GRPO being more widely adopted due to its connection to PPO's clipping mechanism.

Process Reward Models and Step-Level Feedback

Standard RLHF uses outcome reward models (ORMs): the reward is assigned to the entire response after generation. Process Reward Models (PRMs, Lightman et al., 2023)

assign rewards to individual reasoning steps, providing denser supervision for multi-step reasoning tasks.

The PRM objective is: $\mathcal{L}_{\text{PRM}}(\phi) = -\mathbb{E}_{(q,s_1,\ldots,s_N,l_1,\ldots,l_N)}[\sum_i \log \sigma(r_\phi(q, s_{1:i}) \cdot l_i)]$ where s_i are reasoning steps, $l_i \in \{+1, -1\}$ are step-level labels (correct/incorrect), and $r_\phi(q, s_{1:i})$ is the PRM score for step i.

PRMs enable a powerful inference-time strategy: *best-of-N with PRM scoring*—generate N reasoning chains and select the one with the highest PRM score at the final step. This achieves better accuracy than majority voting (best-of-N with majority vote) at the same inference cost, because the PRM can distinguish correct-reasoning-correct-answer from incorrect-reasoning-correct-answer.

Preference Learning Beyond Bradley-Terry

Nash Equilibrium in Preference Optimisation

The Bradley-Terry model assumes a latent scalar reward determines preferences. This is restrictive: human preferences are often *intransitive* (preferring A to B, B to C, but C to A), violating the scalar reward assumption. Nash Learning from Human Feedback (Munos et al., 2023) replaces the scalar reward with a preference function $\mathcal{P}(y, y')$ that can be non-transitive.

The optimal policy under Nash RLHF is the Nash equilibrium of the zero-sum game: $\pi^* = \arg\max_\pi \min_\mu \mathbb{E}_{y\sim\pi, y'\sim\mu}[\mathcal{P}(y, y')]$ This is the mixed strategy Nash equilibrium: no unilateral deviation from π^* can improve the expected win rate against $\mu^* = \pi^*$. The Nash policy π^* cannot be exploited: it wins at least as often as any other policy on average.

Reward Model Uncertainty and Conformal Prediction

A fundamental problem in RLHF: the reward model r_ϕ is trained on a finite dataset $\mathcal{D}$ and may be overconfident outside its training distribution. When the policy is fine-tuned to maximise r_ϕ, it may find responses with high r_ϕ but low true reward r^*—reward hacking.

Conformal prediction (Vovk, Gammerman, and Shafer, 2005) provides distribution-free prediction sets with guaranteed coverage: for any significance level α, the conformal prediction set $\mathcal{C}(x) = \{y : \hat{s}(x, y) \leq q_{1-\alpha}\}$ satisfies $P(y \in \mathcal{C}(x)) \geq 1 - \alpha$ where $q_{1-\alpha}$ is the $(1-\alpha)$-quantile of calibration scores. Applied to RLHF: the conformal prediction set for responses with reward above threshold τ provides a guaranteed lower bound on precision (fraction of set members with true reward $\geq \tau$) under mild exchangeability assumptions.

Cross-Reference to Other Parts. The Nash equilibrium formulation of RLHF connects to the multi-agent RL of Chapter 8 (this part): the preference optimisation game between policy and human is a two-player zero-sum game, and the Nash equilibrium is its solution. The conformal prediction framework connects to the PAC learning theory of Part III: both provide finite-sample guarantees on generalisation without distributional assumptions, using

different mathematical frameworks (probably approximately correct vs. exchangeability).

Memory-Efficient Attention and Long-Context Transformers

Historical and Motivational Context. *Standard transformer attention has $O(L^2)$ memory in sequence length L, limiting context to ∼2048 tokens on 40GB GPUs with standard implementation. FlashAttention (Dao et al., 2022) circumvented this with a tiling algorithm that recomputes attention activations during the backward pass, reducing memory from $O(L^2)$ to $O(L)$ while maintaining exact output. This enabled the scaling of context from 2K to 100K+ tokens, fundamentally changing what language models can process.*

FlashAttention: Tiling and Recomputation

Standard attention $S = QK^T/\sqrt{d}$, $P = \text{softmax}(S)$, $O = PV$ requires materialising $S, P \in \mathbb{R}^{L\times L}$, costing $O(L^2)$ memory.

FlashAttention (Dao et al., 2022) avoids storing the full attention matrix:

1. Partition Q, K, V into blocks of size $B_r \times d$ and $B_c \times d$.
2. For each query block Q_i, iterate over all key-value blocks (K_j, V_j): compute the partial attention $S_{ij} = Q_i K_j^T/\sqrt{d}$, update the running softmax normaliser and output.
3. The running maximum m_i and normaliser ℓ_i enable computing the correct softmax without storing all L scores: $m_i^{(\text{new})} = \max(m_i, \text{rowmax}(S_{ij}))$, $O_i \leftarrow \text{diag}(e^{m_i - m_i^{(\text{new})}})O_i + e^{S_{ij} - m_i^{(\text{new})}} V_j$.

Memory is $O(L)$ (blocks only); compute is still $O(L^2 d)$ but with $5\times$–$8\times$ wall-clock speedup due to reduced HBM reads/writes.

Ring Attention and Sequence Parallelism

Ring Attention (Liu et al., 2023) distributes L across N GPUs: each GPU holds L/N tokens. In each step of the ring, GPUs compute partial attention between their local queries and the current key-value block (received from the previous GPU), then pass the key-value block to the next GPU. After N steps, each GPU has accumulated the full attention output for its local queries.

This enables sequence lengths of $L = N \times L_{\text{local}}$, growing linearly with GPU count. Combined with FlashAttention's $O(L)$ memory per GPU, Ring Attention achieves $O(L/N)$ memory per GPU for any sequence length.

Cross-Reference to Other Parts. FlashAttention's tiling algorithm is related to gradient checkpointing (Part VI, Chapter 5): both trade compute for memory by recomputing activations during the backward pass rather than storing them. The ring attention parallel decomposition is related to the ZeRO-3 parameter sharding strategy (Part VI, Chapter 5): both shard a large tensor (attention matrix / model parameters) across GPUs and use all-

reduce communication to aggregate results.

TD(λ), Eligibility Traces, and the Spectrum of TD Methods

Historical and Motivational Context. *Temporal difference learning (Sutton, 1988) was introduced as a bridge between Monte Carlo methods (which wait until the end of an episode to update, giving unbiased but high-variance returns) and one-step TD (which bootstraps immediately, giving biased but low-variance updates). Eligibility traces (Klopf, 1972; Sutton and Barto, 1981) provided the mechanism for smoothly interpolating between these extremes via $\lambda \in [0, 1]$: $\lambda = 0$ gives one-step TD, $\lambda = 1$ gives full Monte Carlo.*

The λ-Return and the TD(λ) Algorithm

The n-step return provides an intermediate between one-step TD and Monte Carlo:

$$G_t^{(n)} = \sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^n V(s_{t+n}) \tag{24}$$

The λ-return is an exponentially weighted average of n-step returns:

$$G_t^\lambda = (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)} \tag{25}$$

The weight $(1-\lambda)\lambda^{n-1}$ decays geometrically with n, giving most weight to short-term returns (biased but low variance) when λ is small, and spreading weight across longer returns (unbiased but high variance) when λ is large.

Eligibility traces. In the online setting (where we update at every step), the λ-return algorithm is equivalent to maintaining an eligibility trace $e_t(s)$ for each state:

$$\begin{aligned} e_t(s) &= \gamma\lambda\, e_{t-1}(s) + \mathbf{1}[s_t = s] \\ V(s) &\leftarrow V(s) + \alpha\, \delta_t\, e_t(s) \end{aligned} \tag{26}$$

where $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$ is the TD error. The eligibility trace tracks how recently each state was visited, scaling the update by the recency of the state's contribution to the current TD error.

Theorem 12.1 (Equivalence of Forward and Backward Views)**.** *Under offline updates (updating only at episode end), the TD(λ) backward view (using eligibility traces) produces exactly the same weight updates as the forward view (using λ-returns). Under online updates (updating at every step), the two views differ but the backward view is computationally efficient: $O(|S|)$ per step regardless of episode length.*

The Unifying View: TD as Gradient Descent on the VE Loss

Define the value error as $\overline{VE}(\mathbf{w}) = \sum_s \mu(s)[V_\mathbf{w}(s) - V^\pi(s)]^2$ where μ is the on-policy state distribution. The gradient is: $\nabla\overline{VE}(\mathbf{w}) = \sum_s \mu(s)[V_\mathbf{w}(s) - V^\pi(s)]\nabla V_\mathbf{w}(s)$. Since

$V^\pi(s)$ is unknown, TD replaces it with the TD target $y_t = r_t + \gamma V_{\mathbf{w}}(s_{t+1})$—a *semi-gradient* step: $\mathbf{w} \leftarrow \mathbf{w} + \alpha[y_t - V_{\mathbf{w}}(s_t)]\nabla V_{\mathbf{w}}(s_t)$. The term "semi-gradient" reflects that the target y_t also depends on $\mathbf{w}$ through the bootstrap, but this dependence is not differentiated. This is the source of the deadly triad instability: the semi-gradient is not the gradient of any objective, and off-policy semi-gradient TD can diverge.

Actor-Critic Methods and the Natural Policy Gradient

The Actor-Critic Architecture

Actor-critic methods combine a policy (actor) with a value function estimate (critic):

- **Actor:** policy $\pi_\theta(a|s)$ updated by the policy gradient.
- **Critic:** value function $V_w(s)$ updated by TD learning.

The actor uses the critic's estimate to compute the advantage $A^\pi(s,a) \approx r + \gamma V_w(s') - V_w(s)$, reducing variance while introducing bias from the imperfect critic.

A3C (Mnih et al., 2016): Asynchronous actor-critic with multiple parallel workers. Each worker has its own copy of the policy and environment, computes n-step returns, and sends gradients to a central parameter server. The asynchrony breaks correlation between sequential updates and provides exploration diversity.

A2C: Synchronous A3C: all workers synchronise before updating. Simpler to implement and debug; the gradient is averaged over the batch of workers, reducing variance further.

The Natural Policy Gradient

The natural policy gradient (Kakade, 2002) uses the Fisher information metric on the space of policies:

$$\tilde{\nabla} J(\theta) = F(\theta)^{-1} \nabla J(\theta) \tag{27}$$

where $F(\theta) = \mathbb{E}_{\pi_\theta}[\nabla \log \pi_\theta(a|s) \nabla \log \pi_\theta(a|s)^T]$ is the Fisher information matrix. The natural gradient is the steepest ascent direction in the space of distributions equipped with the Fisher metric—it is invariant to reparameterisation of the policy.

> **Cross-Reference to Other Parts.** The natural policy gradient is the RL application of the natural gradient introduced in Part I (Chapter 4) for supervised learning. The Fisher information matrix $F(\theta)$ in the policy space is the same object as the Fisher metric on the statistical manifold; K-FAC (Part I) provides an efficient approximation. TRPO's trust region constraint enforces a fixed-size step in the Fisher metric, making it equivalent to a proximal natural gradient step.

Reward Shaping, Intrinsic Motivation, and Exploration

> **Historical and Motivational Context.** *The exploration-exploitation tradeoff is one of the oldest problems in RL: a rational agent must balance exploring new actions (to discover potentially*

better strategies) against exploiting known good actions. In multi-armed bandits (Robbins, 1952; Lai and Robbins, 1985), the optimal exploration strategy is known: the UCB algorithm achieves $O(\sqrt{T \log T})$ regret after T rounds. In MDPs, exploration is harder because the agent cannot easily estimate the uncertainty of its value estimates in the function approximation setting.

Potential-Based Reward Shaping

Reward shaping modifies the reward function to guide exploration: $r'(s, a, s') = r(s, a, s') + F(s, a, s')$. Ng, Harada, and Russell (1999) showed that if F is a *potential-based* shaping function $F(s, a, s') = \gamma\Phi(s') - \Phi(s)$ for any potential $\Phi : S \to \mathbb{R}$, then the shaped reward r' has the same set of optimal policies as the original reward r.

Theorem 14.1 (Potential-Based Shaping Invariance)**.** *If $F(s, a, s') = \gamma\Phi(s') - \Phi(s)$ (telescoping sum over a trajectory): $\sum_t \gamma^t F(s_t, a_t, s_{t+1}) = \sum_t \gamma^t [\gamma\Phi(s_{t+1}) - \Phi(s_t)] \to -\Phi(s_0)$ This is a constant (independent of the policy trajectory), so the shaped and original optimal policies coincide. Non-potential-based shaping can change the optimal policy, potentially leading to reward hacking.*

Intrinsic Motivation: Count-Based and Prediction-Based Exploration

Count-based exploration (Bellemare et al., 2016) augments the reward with an exploration bonus $r_{\text{int}}(s) = \beta / \sqrt{N(s)}$ where $N(s)$ is the visit count. For high-dimensional state spaces, exact counts are intractable; pseudo-counts from a density model $\hat{\rho}_n$ are used: $\hat{N}(s) = \hat{\rho}_n(s) n / (1 - \hat{\rho}_{n+1}(s) + \hat{\rho}_n(s))$.

Curiosity-driven exploration (Pathak et al., 2017) uses prediction error as a reward signal: $r_{\text{int}}(s_t, a_t, s_{t+1}) = \|\hat{\phi}(s_{t+1}) - \phi(s_{t+1})\|^2$ where $\hat{\phi}(s_{t+1})$ is the predicted next-state embedding and ϕ is a learned feature space invariant to irrelevant visual variations. This incentivises the agent to explore transitions that its world model cannot yet predict accurately.

Multi-Agent RL and Game Theory

Historical and Motivational Context. *Multi-agent RL (MARL) addresses settings where multiple agents interact simultaneously, each learning its own policy. The mathematical foundation is game theory, initiated by von Neumann and Morgenstern (1944) and Nash (1951). Nash's equilibrium concept—a strategy profile where no agent can improve by unilateral deviation—provides the solution concept for multi-agent systems. The AlphaGo self-play algorithm (Silver et al., 2016) and the OpenAI Five (Berner et al., 2019) demonstrated that RL agents trained against themselves via self-play can achieve superhuman performance in complex multi-player games.*

Nash Equilibria and the Minimax Theorem

In a two-player zero-sum game with payoff matrix $A \in \mathbb{R}^{m \times n}$: player 1 chooses a mixed strategy $x \in \Delta_m$ (probability vector over m pure strategies), player 2 chooses $y \in \Delta_n$. The minimax theorem (von Neumann, 1928) states:

$$\max_{x \in \Delta_m} \min_{y \in \Delta_n} x^T A y = \min_{y \in \Delta_n} \max_{x \in \Delta_m} x^T A y = V^* \tag{28}$$

The saddle point (x^*, y^*) satisfying $(x^*)^T A y \geq V^* \geq x^T A y^*$ for all (x, y) is the Nash equilibrium. The value V^* is the game's value.

Self-Play and the AlphaZero Training Regime

AlphaZero (Silver et al., 2018) trains a single network to evaluate positions and generate moves, using self-play to generate training data:

1. **Self-play games:** Monte Carlo Tree Search (MCTS) uses the network to guide search: $\text{UCB}(s,a) = Q(s,a) + c_{\text{puct}}\, \pi_\theta(a|s)\sqrt{N(s)}/(1+N(s,a))$.
2. **Training targets:** $(v, \pi) =$ MCTS result serves as supervision for the network.
3. **Network update:** minimise $\mathcal{L} = (v-z)^2 - \pi^T \log \hat{\pi} + c\|\theta\|^2$ where $z \in \{-1, 0, 1\}$ is the game outcome.

The self-play loop implements a policy iteration procedure: each MCTS-guided policy is strictly better than the network policy, and training the network to imitate MCTS outputs improves the network, which improves MCTS, which improves the network.

Reward Modelling, Scalable Oversight, and Open Problems

Reward Hacking and Goodhart's Law

Goodhart's Law states: "When a measure becomes a target, it ceases to be a good measure." In RL: when a proxy reward r_ϕ is maximised, the policy may find high-reward behaviours that do not correspond to high true reward r^*. This is *reward hacking*.

The KL regularisation $-\beta D_{KL}(\pi \| \pi_{\text{ref}})$ in RLHF limits reward hacking by preventing the policy from deviating too far from the reference. The implicit constraint is that the policy cannot explore regions of the response space not covered by π_{ref}—but this also limits the policy's ability to genuinely improve.

Debate and Amplification

Amplification (Christiano et al., 2018) provides a recursion for scalable oversight: if a human can evaluate a task with assistance from an AI that can answer subquestions, the human can evaluate more complex tasks than they could alone. The recursion $\text{HCH}^0 =$ human, $\text{HCH}^{k+1} =$ "human assisted by HCH^k" converges to a system that can answer any question the human could answer with unlimited time and memory.

Debate (Irving et al., 2018) trains two AI judges to debate: judge 1 argues that its answer is correct, judge 2 argues the opposite. A human referee evaluates the debate. Under the assumption that truth is easier to defend than falsehood (the "lying is hard" assumption), debate converges to truthful answers even for questions the human could not directly evaluate.

> **Cross-Reference to Other Parts.** The reward modelling perspective connects to the causal inference framework of Part VI: reward hacking corresponds to spurious correlation between the proxy reward and the true reward, and IRM-style invariance constraints could

in principle identify reward features that are causally related to human values across environments. The debate framework is a form of adversarial training (Part IV, Chapter 2) applied to the alignment problem: two policies compete, and the competition drives both toward truth.

Inverse Reinforcement Learning and Imitation Learning

Historical and Motivational Context. *Inverse Reinforcement Learning (IRL, Russell, 1998; Ng and Russell, 2000) asks the inverse of the standard RL problem: given a policy (or observed behaviour), infer the reward function that the policy is optimising. The motivation is that it is often easier to demonstrate desired behaviour than to specify a reward function explicitly—a human can show a robot how to make coffee more easily than they can write down all the sub-goals involved. IRL provides the formal tools for inferring reward structure from demonstrations, connecting to the broader alignment problem of recovering human values from observed behaviour.*

Maximum Entropy IRL

Maximum Entropy IRL (Ziebart et al., 2008) resolves the under-determined nature of IRL (many reward functions are consistent with a given policy) by selecting the reward function that makes the observed policy maximum entropy:

$$r^* = \arg\max_r H(\pi_r) \quad \text{s.t.} \quad \mathbb{E}_{\pi_r}[\phi(s,a)] = \mathbb{E}_{\pi^*}[\phi(s,a)] \tag{29}$$

where $\phi(s,a)$ are feature expectations and $H(\pi_r)$ is the entropy of the induced policy. The solution is a Boltzmann distribution over trajectories: $p(\tau) \propto \exp(r(\tau))$, where $r(\tau) = \sum_t r(s_t, a_t)$.

The gradient of the log-likelihood with respect to reward parameters θ:

$$\nabla_\theta \mathcal{L} = \mathbb{E}_{\pi^*}[\phi] - \mathbb{E}_{p(\tau|\theta)}[\phi] = \mu_{\text{expert}} - \mu_{\text{policy}} \tag{30}$$

is the difference between expert feature expectations and model feature expectations—exactly the moment-matching condition. Maximising the log-likelihood of expert demonstrations drives the model's feature expectations to match the expert's.

Generative Adversarial Imitation Learning (GAIL)

GAIL (Ho and Ermon, 2016) connects IRL and GANs: instead of explicitly recovering a reward function, it trains a discriminator $D_w(s,a)$ to distinguish expert state-action pairs from policy state-action pairs, and uses the discriminator's output as the reward:

$$r(s,a) = -\log(1 - D_w(s,a)) \tag{31}$$

The policy is updated with PPO or TRPO to maximise this reward, and the discriminator is updated to distinguish expert from policy behaviour: $\max_w \mathbb{E}_{\pi^*}[\log D_w(s,a)] + \mathbb{E}_{\pi}[\log(1 - D_w(s,a))]$. GAIL is related to the original GAN by the substitution: generator $\equiv$ policy, real data $\equiv$ expert demonstrations, fake data $\equiv$ policy trajectories.

Mathematics of Deep Learning — Part I: The Geometric Foundations

Cross-Reference to Other Parts. GAIL is the RL analogue of the GAN training framework of Part IV (Chapter 2): the discriminator plays the role of the reward model, the policy plays the role of the generator, and the adversarial training loop drives the policy distribution toward the expert distribution. The theoretical connection is exact: GAIL with a JS-divergence discriminator minimises $\mathrm{JSD}(\rho_{\pi^*} \| \rho_\pi)$ between the state-action occupancy measures, analogous to the GAN minimising JSD between real and generated image distributions.

Constrained Markov Decision Processes

Constrained MDPs (CMDPs) extend the standard MDP with budget constraints on auxiliary costs. The CMDP objective:

$$\max_{\pi} J(\pi) \quad \text{s.t.} \quad J_k^c(\pi) \leq d_k, \quad k = 1, \dots, K \tag{32}$$

where $J_k^c(\pi) = \mathbb{E}_\pi[\sum_t \gamma^t c_k(s_t, a_t)]$ is the expected cumulative cost under constraint k and d_k is the budget.

The Lagrangian relaxation: $\mathcal{L}(\pi, \lambda) = J(\pi) - \sum_k \lambda_k (J_k^c(\pi) - d_k)$ converts the constrained problem into an unconstrained min-max: $\max_\pi \min_{\lambda \geq 0} \mathcal{L}(\pi, \lambda)$. Under strong duality (Slater's condition: $\exists \pi$ with $J_k^c(\pi) < d_k$), the primal and dual optima coincide.

Primal-dual policy gradient (Paternain et al., 2019) alternates:

$$\begin{aligned} \pi_{t+1} &\leftarrow \pi_t + \alpha \nabla_\pi \mathcal{L}(\pi_t, \lambda_t) \\ \lambda_{t+1} &\leftarrow [\lambda_t - \beta (d_k - J_k^c(\pi_t))]_+ \end{aligned} \tag{33}$$

The dual variable λ_k increases when the constraint is violated (cost exceeds budget) and decreases when slack is available, acting as a dynamic penalty that enforces the constraint in expectation.

Optimism in the Face of Uncertainty: Exploration in MDPs

RMAX and the Near-Optimal Exploration Theorem

RMAX (Brafman and Tennenholtz, 2002) formalises the exploration strategy "optimism in the face of uncertainty": unknown states are assigned the maximum possible reward $r_{\max}$, incentivising exploration.

Theorem 19.1 (RMAX Near-Optimality)**.** *RMAX reaches an ε-optimal policy with probability $\geq 1 - \delta$ after at most $O(|S|^2|A| \frac{r_{max}^2}{(1-\gamma)^6 \varepsilon^2} \log \frac{|S||A|}{\delta})$ steps. The key property: RMAX is* PAC-MDP*—it attains near-optimal behaviour after polynomially many steps with high probability, resolving the exploration problem in polynomial time for tabular MDPs.*

Posterior Sampling for Reinforcement Learning

PSRL (Osband et al., 2013) maintains a Bayesian posterior over MDPs and at each episode samples a single MDP from this posterior, then acts optimally with respect

to the sampled MDP. This Thompson Sampling approach achieves a Bayesian regret bound of $\tilde{O}(\tau S\sqrt{AT})$ over T steps, where τ is the diameter of the MDP and S, A are state and action counts.

The key intuition: PSRL naturally balances exploration and exploitation because

- When the posterior is diffuse (high uncertainty), the sampled MDP may be very different from the true one—but acting optimally for the sampled MDP explores uncertain regions.
- As evidence accumulates, the posterior concentrates around the true MDP, and acting optimally for the sampled MDP approaches acting optimally for the true MDP.

Safe Reinforcement Learning and the CVar Objective

Risk-Sensitive Objectives

Standard RL maximises expected return $\mathbb{E}[\sum_t r_t]$, which is risk-neutral: a policy that achieves $+100$ with probability 0.99 and -1000 with probability 0.01 has the same expected value as one that achieves -1 deterministically. Safe RL replaces the mean with a risk measure:

Conditional Value at Risk (CVaR): $\text{CVaR}_\alpha(Z) = \frac{1}{\alpha}\int_0^\alpha \text{VaR}_u(Z)\,du = \mathbb{E}[Z|Z \le \text{VaR}_\alpha(Z)]$ — the expected return in the worst α-fraction of outcomes.

A CVaR-constrained policy $\pi^* = \arg\max_\pi \text{CVaR}_\alpha(Z^\pi)$ avoids catastrophic outcomes that a mean-maximising policy would accept. The dual representation: $\text{CVaR}_\alpha(Z) = \min_{q:\mathbb{E}[q]=1,\, 0\le q\le 1/\alpha} \mathbb{E}[qZ]$ expresses CVaR as a worst-case expectation over a set of distorted probability measures, connecting risk-sensitive RL to distributionally robust optimisation.

Lyapunov-Based Safety Constraints

A Lyapunov function $L : S \to \mathbb{R}_{\ge 0}$ provides safety certificates for dynamical systems. If $L(s) = 0$ iff s is safe and $\mathbb{E}[L(s_{t+1})|s_t, a_t] \le \gamma L(s_t) + c$ for all (s_t, a_t), then the system is guaranteed to return to the safe set.

Safe RL via Lyapunov functions (Berkenkamp et al., 2017; Chow et al., 2019) enforces: $\nabla L(s)^T f_\pi(s) \le -\epsilon L(s)$ (continuous time) $\mathbb{E}[L(s_{t+1})|s_t, a_t] \le (1-\epsilon)L(s_t)$ (discrete time) as a constraint on the policy, guaranteeing that the Lyapunov function decreases in expectation—a formal certificate of safety.

Language Model Reasoning: Chain-of-Thought Mathematics

Why CoT Works: The Scratchpad Complexity Argument

Standard transformers with depth D and width W compute functions in the circuit complexity class TC^0 (constant-depth threshold circuits). The class TC^0 cannot compute parity, majority vote on n bits, or iterated multiplication—all requiring super-constant

depth.

When CoT adds k intermediate tokens, the effective computational depth increases to $O(kD)$: each intermediate token is computed by a depth-D transformer, but the next token sees all previous intermediate computations. Feng et al. (2023) prove:

Theorem 21.1 (CoT Computational Power)**.** *A D-layer, W-width transformer with k CoT tokens can compute any function in NC^1 (log-depth circuits) when $k = O(\log n)$ and $D = O(1), W = O(n^{O(1)})$. Without CoT ($k = 0$), the transformer is limited to TC^0, which is strictly contained in NC^1 under standard complexity assumptions.*

The practical consequence: arithmetic (addition, multiplication), counting, and sorting all require $\Omega(\log n)$ CoT steps for n-digit inputs. This explains why models fail at multi-digit arithmetic without CoT but succeed with step-by-step reasoning.

Self-Consistency Decoding

Self-consistency (Wang et al., 2023) generates N independent CoT reasoning chains and takes the majority vote over final answers: $\hat{y} = \text{MajVote}(\{y_i\}_{i=1}^N)$. Self-consistency improves over greedy decoding by a margin that grows with N (approximately as $O(\sqrt{N})$ for binomial vote aggregation) and with task difficulty.

The theoretical justification: if each reasoning chain is independently correct with probability $p > 0.5$, the majority vote is correct with probability $P(\text{Binomial}(N, p) > N/2) \to 1$ as $N \to \infty$ by the law of large numbers. For $p = 0.7$ and $N = 8$: $P(\text{correct}) \approx 0.97$ versus 0.70 for single-chain greedy decoding.

Offline RL: Learning from Fixed Datasets

Historical and Motivational Context. *Offline reinforcement learning (also called batch RL or offline policy optimisation) addresses the setting where the agent cannot interact with the environment: it must learn the best possible policy from a fixed dataset of transitions $\mathcal{D} = \{(s, a, r, s')\}$ collected by some behaviour policy μ. This is the dominant setting in many real-world applications (healthcare, robotics, autonomous driving), where online exploration is unsafe or expensive. The fundamental challenge is distributional shift: the Q-function is evaluated at out-of-distribution (s, a) pairs, leading to Q-value overestimation for actions not well-covered by $\mathcal{D}$.*

Conservative Q-Learning (CQL)

CQL (Kumar et al., 2020) penalises Q-values at out-of-distribution actions:

$$\mathcal{L}_{\text{CQL}}(Q) = \alpha \underbrace{\mathbb{E}_{s\sim\mathcal{D}}[\log \sum_a \exp Q(s,a) - \mathbb{E}_{a\sim\hat{\pi}_\beta}[Q(s,a)]]}_{\text{conservation penalty}} + \underbrace{\mathcal{L}_{\text{TD}}(Q)}_{\text{Bellman error}} \tag{34}$$

The conservation penalty pushes down Q-values for all actions (via the log-sum-exp term) while pushing up Q-values for actions in the dataset (via the behaviour policy $\hat{\pi}_\beta$). This produces a Q-function that is a *lower bound* on the true Q-function, ensuring that the learned policy only selects actions with conservatively estimated values.

Theorem 22.1 (CQL Lower Bound Property)**.** *Under mild regularity conditions, CQL with sufficiently large α produces a Q-function $\hat{Q}$ satisfying $\hat{Q}(s,a) \leq Q^{\pi^*}(s,a)$ for all (s,a) with high probability, ensuring that the policy derived from $\hat{Q}$ does not exploit overestimated Q-values for out-of-distribution actions.*

Decision Transformer: RL as Sequence Modelling

Decision Transformer (Chen et al., 2021) recasts offline RL as conditional sequence modelling: given a desired *return-to-go* $\hat{R}_t = \sum_{k=t}^{T} r_k$ and past context $(R_1, s_1, a_1, \ldots, R_{t-1}, s_{t-1}, a_{t-1})$, predict the next action a_t. The model is a GPT-style transformer trained on offline trajectories.

At inference time, condition on a desired high return (e.g. $\hat{R}_0 = R_{\max}$) and generate actions autoregressively. This sidesteps the Bellman equation entirely: no value function, no bootstrapping, no distributional shift problem. The tradeoff: Decision Transformer cannot exceed the performance seen in its offline dataset (it learns to imitate trajectories, not to improve on them), while CQL can in principle extrapolate to policies better than any in the dataset.

Concluding Remarks on Part V

Reinforcement learning and alignment present the deepest mathematical challenges in machine learning: the credit assignment problem, the exploration–exploitation tradeoff, the instability of off-policy learning with function approximation, and the misspecification problem in reward learning. The tools developed here—from Bellman contraction theorems to the DPO reparameterisation—provide rigorous foundations for systems that must act wisely in uncertain environments and in accordance with human values.

The alignment problem, in its modern form, reduces to learning a distribution over human preferences that is rich enough to guide a capable system through novel situations. The gap between current RLHF methods (which optimise a proxy reward model) and true alignment (which requires the system to understand human values in their full generality) remains the central open problem at the intersection of mathematics, computer science, and philosophy.

The mechanistic interpretability programme closes the circle by providing tools to verify, post-hoc, that learned systems have implemented safe and interpretable algorithms—connecting the statistical learning theory of Part III with the architectural analysis of Part II.

Appendix: Key Formulae Reference

A.1 MDP and Bellman Equations

Bellman consistency: $V^\pi(s) = \sum_a \pi(a|s)[R(s,a) + \gamma \sum_{s'} P(s'|s,a)V^\pi(s')]$

Bellman optimality: $V^*(s) = \max_a [R(s,a) + \gamma \sum_{s'} P(s'|s,a)V^*(s')]$

Q-learning update: $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)]$

TD(λ) eligibility trace: $e_t(s) = \gamma\lambda\, e_{t-1}(s) + \mathbf{1}[s_t = s], \quad V(s) \leftarrow V(s) + \alpha\delta_t e_t(s)$

A.2 Policy Gradient Methods

Policy Gradient Theorem: $\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[\sum_t \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot A^\pi(s_t, a_t)]$

PPO clipped objective: $\mathcal{L}_{\text{CLIP}}(\theta) = \mathbb{E}_t[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t)]$

GAE advantage: $\hat{A}_t = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta^V_{t+l}, \quad \delta^V_t = r_t + \gamma V(s_{t+1}) - V(s_t)$

MaxEnt optimal policy: $\pi^*(a|s) \propto \exp(Q^*_{\text{soft}}(s, a)/\alpha)$

A.3 RLHF and Alignment

Bradley-Terry model: $p^*(y_w \succ y_l|x) = \sigma(r^*(x, y_w) - r^*(x, y_l))$

Optimal RLHF policy: $\pi^*(y|x) = \pi_{\text{ref}}(y|x) \exp(r(x, y)/\beta)/Z(x)$

DPO loss: $\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}[\log \sigma(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)})]$

DPO implicit reward: $\hat{r}_\theta(x, y) = \beta \log(\pi_\theta(y|x)/\pi_{\text{ref}}(y|x))$

A.4 Model-Based RL

MBPO return bound: $|\eta[\pi] - \eta_{\text{model}}[\pi]| \leq \frac{2\gamma(\gamma^k \varepsilon_m + (1-\gamma^k)(\varepsilon_m + \varepsilon_\pi))}{(1-\gamma)^2}$

Dreamer world model ELBO: $\mathcal{L}_{\text{model}} = \mathbb{E}_q[\sum_t \log p(o_t|h_t, z_t) - \beta D_{KL}(q(z_t|h_t, o_t) \| p(z_t|h_t))]$

A.5 Mechanistic Interpretability

Superposition capacity (sparse features, $p \to 0$): $\mathbb{E}[\text{loss}] = \sum_i (1 - \|u_i\|^2)^2 I_i + \sum_{i\neq j} (u_i \cdot u_j)^2 I_i I_j p$

Sparse autoencoder loss: $\mathcal{L}_{\text{SAE}} = \|h - D\,\text{ReLU}(Eh + b)\|^2 + \lambda \|\,\text{ReLU}(Eh + b)\|_1$

ICL as Bayes: $p_{LM}(y|\text{context}) \approx \mathbb{E}_{C\sim p(C|\text{context})}[p(y|x_{k+1}, C)]$

MATHEMATICS OF DEEP LEARNING

PART VI

ADVANCED MATHEMATICAL STRUCTURES

Statistical Physics · Algebraic Topology · Causal Inference

Scaling Laws · Fine-Tuning at Scale · Multimodal Models

Eight-Part Monograph Series on the Mathematical Principles of Neural Networks

Independent of Companion Volumes — Complete in Itself

Silvano Salvador

© Mathematics of Deep Learning Series

PART VI

ADVANCED MATHEMATICAL STRUCTURES

Statistical Physics · Algebraic Topology · Causal Inference

Scaling Laws · Fine-Tuning at Scale · Multimodal Models

© Mathematics of Deep Learning Series

Silvano Salvador

Introduction: Why Advanced Mathematics Is Necessary

Historical and Motivational Context. *Deep learning became a field of rigorous mathematical inquiry only gradually. The first decade (2010–2020) was dominated by empirical advances with post-hoc theoretical justifications. The universal approximation theorem (Hornik et al., 1989) established that deep networks could represent any continuous function, but said nothing about whether gradient descent would find good representations. Dropout, batch normalisation, and residual connections were introduced empirically and justified theoretically only years later. The phenomena that most sharply challenged the existing theory were: (1) the* generalisation paradox*—overparameterised networks with millions more parameters than training examples generalise well, contradicting the classical wisdom that overfitting requires regularisation; (2)* double descent*—test error follows a non-monotone U-then-U curve as model capacity increases, with a second descent into low error in the overparameterised regime; (3)* grokking*—delayed generalisation where networks first memorise training data and then reorganise into generalisable solutions; (4)* emergent capabilities*—abilities absent at smaller scale that appear abruptly at larger scale, resembling phase transitions. Each of these phenomena required tools from statistical physics, algebraic topology, and information theory that were not part of the standard ML toolkit.*

This part assembles the advanced mathematical frameworks that provide the deepest understanding of why deep learning works and where its limits lie. Each chapter draws on a distinct mathematical tradition, applying it to foundational questions about generalisation, scale, adaptation, and multimodal alignment.

Cross-Reference to Other Parts. The statistical physics of Chapter 1 connects directly to the random matrix theory that underlies the NTK analysis of Part III and the spectral properties of weight matrices studied in Part I. The algebraic topology of Chapter 2 provides invariants of the data manifolds whose existence was posited in Part I and Part II. The causal inference of Chapter 3 addresses the distributional robustness question that classical PAC theory (Part III) cannot answer. The scaling laws of Chapter 4 empirically characterise the sample complexity bounds that PAC-Bayes theory (Part III) predicts.

Statistical Physics of Learning: Spin Glasses, Replicas, and Free Probability

Historical and Motivational Context. *The connection between neural networks and statistical mechanics dates to the Hopfield network (1982), which was explicitly modelled on the Ising spin system. The physicist's toolkit—partition functions, free energy, replicas, mean field theory—was applied to neural networks by Amit, Gutfreund, and Sompolinsky (1985), who derived the storage capacity of Hopfield networks via the replica method. The subsequent development of the statistical physics of learning (Gardner and Derrida, 1988; Seung, Sompolinsky, and Tishby, 1992) provided the first non-asymptotic learning theory from a physics perspective. The modern revival (Advani and Saxe, 2017; Spigler et al., 2019; Geiger et al., 2020) applies these methods to deep networks and the double descent phenomenon, providing a precise explanation of the overparameterisation benefit in terms of the jamming transition.*

The Hopfield Network: Energy Landscape and Phase Transitions

Classical Hopfield networks (Hopfield, 1982) store M patterns $\{\xi^\mu\}_{\mu=1}^M$ in a weight matrix $W = \frac{1}{n}\sum_\mu \xi^\mu(\xi^\mu)^T$ with energy $E(x) = -\frac{1}{2}x^T W x$. Update dynamics $x_i \leftarrow \text{sign}(\sum_j W_{ij}x_j)$ are gradient descent on E, converging to attractors (local minima).

The Amit–Gutfreund–Sompolinsky (AGS) analysis yields a critical capacity $\alpha_c = M/n \approx 0.138$:

- Below α_c: each pattern ξ^μ is a stable attractor with basin of attraction containing all states within Hamming distance $0.1n$ of ξ^μ. Retrieval proceeds correctly from corrupted inputs.
- Above α_c: the network undergoes a first-order phase transition to a *spin glass* phase with exponentially many metastable states at low energy, none corresponding to stored patterns. Retrieval fails catastrophically.

Theorem 1.1 (AGS Capacity via Replica Method)**.** *For a Hopfield network with n neurons and $M = \alpha n$ random ± 1 patterns, the replica-symmetric free energy per neuron is:*

$$f = \text{extr}_{m,q,r}\left\{-\frac{1}{2}\frac{\alpha(1-q)^2}{(1-\alpha(1-q))^2} + \textit{entropy terms}\right\} \tag{1}$$

The critical load $\alpha_c \approx 0.138$ is determined by the Almeida–Thouless instability condition, where the replica-symmetric solution develops a negative eigenvalue in the Hessian of the free energy.

The Replica Method: Deriving the Quenched Free Energy

The partition function $Z = \sum_x \exp(-\beta E(x))$ of a disordered system (with random patterns ξ) is itself random. The *quenched* free energy $F = -(1/\beta)\mathbb{E}_\xi[\log Z]$ requires computing the expectation of a logarithm—intractable directly. The replica trick uses:

$$\mathbb{E}_\xi[\log Z] = \lim_{n\to 0}\frac{\mathbb{E}_\xi[Z^n]-1}{n} \tag{2}$$

For integer n, $\mathbb{E}_\xi[Z^n]$ is the partition function of n replicas of the system with the same disorder ξ, hence tractable:

$$\mathbb{E}_\xi[Z^n] = \sum_{x^1,\dots,x^n} \mathbb{E}_\xi\left[\exp\left(-\beta \sum_a E(x^a)\right)\right] \tag{3}$$

The disorder average couples the replicas through the overlap matrix $q_{ab} = \frac{1}{n}\sum_i x_i^a x_i^b$. Under the replica-symmetric (RS) ansatz, $q_{ab} = q$ for all $a \neq b$, the saddle-point equations become:

$$\begin{aligned} m &= \int Dz \, \tanh(\beta(mJ + z\sqrt{\alpha q J^2/(1-\alpha)})) \\ q &= \int Dz \, \tanh^2(\beta(mJ + z\sqrt{\alpha q J^2/(1-\alpha)})) \end{aligned} \tag{4}$$

where $Dz = (2\pi)^{-1/2} e^{-z^2/2} dz$ is the Gaussian measure and J is the interaction strength. The RS ansatz breaks down in the spin glass phase, where Parisi's full replica symmetry breaking (RSB) scheme is required.

Free Probability and the Marchenko-Pastur Law

Free probability (Voiculescu, 1991) is the non-commutative analogue of classical probability theory. For large random matrices, free probability provides exact eigenvalue distributions.

The Marchenko–Pastur law governs the eigenvalue distribution of the sample covariance matrix $S = \frac{1}{n} X^T X$ for $X \in \mathbb{R}^{n \times p}$ with i.i.d. entries of variance σ^2, as $n, p \to \infty$ with $p/n \to \gamma$:

$$\mu_{MP}(\lambda) = \frac{\sqrt{(\lambda_+ - \lambda)(\lambda - \lambda_-)}}{2\pi\sigma^2\gamma\lambda} \mathbf{1}_{[\lambda_-, \lambda_+]} \tag{5}$$

where $\lambda_\pm = \sigma^2(1 \pm \sqrt{\gamma})^2$. This law describes the spectrum of weight matrix products in neural networks at initialisation and provides a baseline for detecting structure added by training.

The R-transform $R_A(z)$ of a random matrix A satisfies: $R_{A+B}(z) = R_A(z) + R_B(z)$ for free matrices A, B. This enables exact computation of the spectral density of sums of random matrices—directly relevant to understanding weight updates: if ΔW is free from the pre-trained weight W_0, the spectrum of $W = W_0 + \Delta W$ is the free convolution of the spectra of W_0 and ΔW.

Cross-Reference to Other Parts. The Marchenko–Pastur law is the bulk spectral distribution governing the NTK eigenvalue spectrum in the infinite-width limit (Part III). The spectral radius of weight matrices, controlled by spectral normalisation (Part IV, Chapter 2), determines stability in the sense of free probability: the Lipschitz constant of the composed network is bounded by the product of spectral radii, which free probability allows to be tracked through training.

The Jamming Transition and Loss Landscape Geometry

The jamming transition—the transition from a flowing to a rigid amorphous solid as density increases—has a precise analogue in neural network training.

Definition 1.2 (Jamming Phases)**.** A network with P parameters is in the *jamming-above* phase when $P > N$ (parameters exceed constraints), and in the *jamming-below* phase when $P < N$. In the jamming-above phase, the loss landscape has no barriers to zero training loss; in the jamming-below phase, perfect fit is geometrically impossible.

Sagun et al. (2017) show that the Hessian spectrum of overparameterised networks has a large bulk near zero (flat directions) plus a small number of outlier eigenvalues (informative curvature). The bulk magnitude is $O(N/P)$, vanishing as overparameterisation increases. This explains why SGD-trained overparameterised networks generalise: SGD noise selects among the continuum of global minima, preferring wide flat regions—large ε-balls with small loss—that correspond to good generalisation by PAC-Bayes bounds (Part III).

The *double descent* phenomenon (Belkin et al., 2019; Nakkiran et al., 2020) is the empirical manifestation of the jamming transition: test error follows a non-monotone curve as model size increases, with a peak at the interpolation threshold $P = N$ (where the network just barely fits the training data and there is no flat-minimum selection) and a second descent as $P \gg N$ (where the jamming-above phase allows implicit regularisation).

Algebraic Topology Applied to Machine Learning

Historical and Motivational Context. *Persistent homology was introduced by Edelsbrunner, Letscher, and Zomorodian (2002) and developed into a theory of stability by Cohen-Steiner, Edelsbrunner, and Harer (2007). Its application to machine learning began with the study of data manifolds (Carlsson, 2009; Singh et al., 2007) and evolved into a family of topologically regularised learning methods (Hu et al., 2019; Hofer et al., 2020). The grokking phenomenon (Power et al., 2022) provided one of the most striking examples of topology guiding machine learning insight: the delayed generalisation transition has a precise topological signature in the activation geometry of the network, measurable by persistent homology of the representation space.*

Persistent Homology: From Point Clouds to Persistence Diagrams

Given a finite point cloud $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^d$, the Čech complex $\check{C}_\varepsilon$ at scale ε has the same homotopy type as the union of ε-balls $\bigcup_i B_\varepsilon(x_i)$. As ε increases from 0 to ∞, topological features are born and die.

Theorem 2.1 (Persistence Structure Theorem)**.** *The k-th persistent homology of a filtration $\{X_\varepsilon\}_{\varepsilon \geq 0}$ decomposes uniquely over a field $\Bbbk$ as a direct sum of interval modules:*

$$H_k(\{X_\varepsilon\}) \cong \bigoplus_i \Bbbk[b_i, d_i) \tag{6}$$

Each interval $[b_i, d_i)$ records the birth ($\varepsilon = b_i$) and death ($\varepsilon = d_i$) of a k-dimensional topological feature. The persistence diagram $\mathcal{D} = \{(b_i, d_i)\}$ is a multiset in $\mathbb{R}^2$. Features near

the diagonal ($d_i - b_i \approx 0$) *are topological noise; features far from the diagonal are genuine topological structure.*

The Wasserstein distance between persistence diagrams: $W_p(\mathcal{D}_1, \mathcal{D}_2) = \inf_\phi (\sum_i \|p_i - \phi(p_i)\|_\infty^p)^{1/p}$ provides a metric on the space of topological summaries. The stability theorem (Cohen-Steiner et al., 2007) guarantees: $W_\infty(\mathcal{D}_f, \mathcal{D}_g) \leq \|f - g\|_\infty$, making persistence diagrams robust to small perturbations of the data.

Application to neural networks. The activation manifold of a trained network—the image of the data manifold under the penultimate-layer embedding—carries topological information about the learned representation. Topological regularisation adds a persistence-based term to the loss: $\mathcal{L}_{\text{topo}} = \lambda \sum_{(b,d)\in\mathcal{D}} (d-b)^2$ (penalising topological noise) or $\lambda \sum_{(b,d)\in\mathcal{D}_{\text{target}}} \min_{(b',d')\in\mathcal{D}} \|(b,d) - (b',d')\|^2$ (matching a target topology).

Sheaf Theory and the Connection Laplacian

A cellular sheaf $\mathcal{F}$ on a graph $G = (V, E)$ assigns a vector space $\mathcal{F}(v)$ to each vertex v and $\mathcal{F}(e)$ to each edge e, together with linear restriction maps $\mathcal{F}_{v \trianglelefteq e} : \mathcal{F}(v) \to \mathcal{F}(e)$.

The *sheaf Laplacian* $\Delta_{\mathcal{F}} = B^T B$, where B is the coboundary operator of the sheaf, generalises the graph Laplacian. For a constant sheaf (all restriction maps are identity), $\Delta_{\mathcal{F}}$ reduces to the standard graph Laplacian. Global sections $H^0(G, \mathcal{F}) = \ker(\Delta_{\mathcal{F}})$ are globally consistent assignments.

Sheaf neural networks (Hansen and Ghrist, 2020) learn the restriction maps $\mathcal{F}_{v \trianglelefteq e}$ from data, enabling message passing that respects non-trivial local agreement structures. This generalises standard graph neural networks (where messages are scalar or vector, with trivial restriction maps) to messages with orientation and direction.

Grokking: Delayed Generalisation as a Topological Phase Transition

Grokking (Power et al., 2022) refers to the phenomenon where a network first memorises the training set and then, after extended training, generalises. The transition is separated by tens of thousands of gradient steps and occurs long after training loss has converged to zero.

Nanda et al. (2023) provide a mechanistic account for modular addition: the network learns a Fourier representation implementing $a + b \mod p$ via circular embeddings $(\cos(2\pi ka/p), \sin(2\pi ka/p))$ composed with the product-to-sum identity $\cos\theta\cos\phi = \frac{1}{2}[\cos(\theta-\phi) + \cos(\theta+\phi)]$.

The grokking transition has a precise topological signature: the persistence diagram of the activation geometry transforms from a scattered, high-noise structure (memorisation) to a circular structure with one prominent H_1 generator at the correct frequency (generalisation). The transition occurs when the L^2 norm of the Fourier solution falls below the norm of the memorisation solution under weight decay—a thermodynamic argument: $\lambda\|W_{\text{Fourier}}\|^2 < \lambda\|W_{\text{memo}}\|^2$ determines when the Fourier solution becomes energetically preferred.

Cross-Reference to Other Parts. The circular representation discovered in grokking is a concrete instance of the geometric structure studied in Part I: the embedding of $\mathbb{Z}/p\mathbb{Z}$ into a circle in $\mathbb{R}^2$ is an isometric embedding of the cyclic group with its word metric into Euclidean space. The topological regularisation perspective connects to the representation theory of Part II: the Fourier decomposition of $\mathbb{Z}/p\mathbb{Z}$ produces the irreducible representations of the cyclic group, which are precisely the complex exponentials.

Causal Inference and Out-of-Distribution Generalisation

Historical and Motivational Context. *The distinction between correlation and causation was central to the debates of early statistics (Fisher vs. Neyman-Pearson). The causal revolution (Pearl, 2000; Peters et al., 2017) provided a formal language—structural causal models and the do-calculus—for reasoning about interventions and counterfactuals. The application to machine learning was motivated by the failure of learned models under distributional shift: models that achieve 99% accuracy on ImageNet can fail catastrophically on data collected from a different hospital, season, or geographic region, not because the task changed but because spurious correlations in training data no longer hold. IRM (Arjovsky et al., 2019) and its successors seek representations that are causally, rather than merely correlationally, related to the label.*

Structural Causal Models and the Do-Calculus

A Structural Causal Model (SCM) consists of a DAG G over variables $X_1, \ldots, X_n$ and structural equations $X_i = f_i(\mathrm{PA}(X_i), U_i)$, where $\mathrm{PA}(X_i)$ are the parents of X_i in G and U_i are independent exogenous noise variables.

The *do-operator* $\mathrm{do}(X_i = x)$ replaces the structural equation for X_i with the constant x, cutting all incoming edges to X_i. The interventional distribution $p(X_j | \,\mathrm{do}(X_i = x))$ answers: what would X_j be if we *set* $X_i = x$ (rather than observing $X_i = x$)?

The key distinction from conditioning: $p(Y|X = x) \neq p(Y|\,\mathrm{do}(X = x))$ when there exist unobserved confounders—hidden common causes of X and Y that create spurious correlations between them.

Theorem 3.1 (Pearl's Do-Calculus Completeness)**.** *The three rules of the do-calculus (applied to the causal graph G and its c-components) are complete for identifying causal effects: $P(Y|\,\mathrm{do}(X))$ is identifiable from observational data if and only if it can be derived using the do-calculus rules. Non-identifiability occurs when the graph has unobserved confounders creating backdoor paths from X to Y that cannot be blocked by conditioning on observed variables.*

Invariant Risk Minimisation

IRM (Arjovsky et al., 2019) seeks a feature representation Φ such that the optimal linear classifier on top of Φ is the same across all training environments $\{e \in \mathcal{E}_{\text{train}}\}$:

$$\min_{\Phi, w} \sum_{e} R^e(w \circ \Phi) \quad \text{s.t.} \quad w \in \arg\min_{w'} R^e(w' \circ \Phi) \quad \forall e \in \mathcal{E}_{\text{train}} \tag{7}$$

The invariance constraint expresses the causal hypothesis that the features used by Φ are causally related to Y—stable across environments—rather than merely correlated

in training data.

The practical relaxation IRM-v1 adds a gradient penalty: $\mathcal{L}_{\text{IRM}}(\Phi) = \sum_e R^e(\mathbf{1} \circ \Phi) + \lambda \sum_e \|\nabla_{w|w=\mathbf{1}} R^e(w \circ \Phi)\|^2$ enforcing that the dummy classifier $w = \mathbf{1}$ is approximately optimal in each environment (meaning Φ already encodes the invariant features).

Distributionally Robust Optimisation

DRO (Sagawa et al., 2020) minimises the worst-group risk: $\min_\theta \max_{g \in \mathcal{G}} R^g(\theta)$. The Lagrangian formulation gives:

$$\min_\theta \max_{q \in \Delta(\mathcal{G})} \sum_g q_g R^g(\theta) = \min_\theta \max_{P' \in \mathcal{P}} \mathbb{E}_{P'}[\ell(f_\theta(x), y)] \tag{8}$$

where $\mathcal{P}$ is the uncertainty set. With χ^2 uncertainty sets ($D_{\chi^2}(P' \| P) \leq \varepsilon$), the dual problem involves the conditional variance of the loss, and empirical optimisation up-weights high-loss examples.

The connection between IRM and DRO: IRM exploits causal structure (known environmental labels) to find invariant features; DRO provides adversarial robustness to any shift within $\mathcal{P}$ without assuming environmental labels. Both methods improve on ERM under covariate shift but face the challenge that the test distribution may lie outside both the causal invariance set and the DRO uncertainty set.

Cross-Reference to Other Parts. The causal model perspective connects to the generalisation theory of Part III: the PAC-Bayes bounds assume that train and test distributions are identical, while IRM and DRO bound the gap under distributional shift in terms of the environmental diversity of training. The do-calculus framework connects to the counterfactual reasoning required for reward shaping in RL (Part V): "what would the reward have been if the agent had taken a different action?" is a counterfactual query in the SCM of the environment.

Scaling Laws and Emergent Capabilities

Historical and Motivational Context. *Power laws in natural phenomena have been documented since Zipf (1935) for word frequencies and Pareto (1896) for wealth distributions. Their appearance in neural network training loss as a function of scale (Hestness et al., 2017; Rosenfeld et al., 2019) was at first surprising: why should a loss function follow a clean power law in model size? The Kaplan et al. (2020) paper provided a systematic study across five orders of magnitude in compute, establishing the power law exponents empirically. The subsequent Chinchilla (Hoffmann et al., 2022) analysis revealed a critical error in the Kaplan prescription—that models should be scaled faster than data—by deriving the optimal allocation analytically and verifying it empirically.*

The Kaplan and Chinchilla Scaling Laws: Full Derivation

The loss decomposition for a language model:

$$L(N, D) = E + \frac{A}{N^\alpha} + \frac{B}{D^\beta} \tag{9}$$

where E is the irreducible entropy of natural language, A/N^α is the underfitting term (too few parameters), and B/D^β is the undertraining term (too few tokens). Empirically, $\alpha \approx 0.076$ and $\beta \approx 0.095$ (Kaplan et al., 2020).

Theorem 4.1 (Optimal Compute Allocation (Chinchilla))**.** *Under the loss decomposition $L(N, D) = E + AN^{-\alpha} + BD^{-\beta}$ and compute constraint $C = 6ND$ (the 6 counts multiply-accumulates per token per parameter in the forward pass, doubled for the backward pass), the compute-optimal allocation satisfies:*

$$N^*(C) = \left(\frac{\alpha A}{\beta B}\right)^{1/(\alpha+\beta)} C^{\beta/(\alpha+\beta)}$$
$$D^*(C) = \left(\frac{\beta B}{\alpha A}\right)^{1/(\alpha+\beta)} C^{\alpha/(\alpha+\beta)} \tag{10}$$

Derivation. We minimise $L(N, C/(6N))$ over N. Setting $\frac{dL}{dN} = -\alpha A N^{-\alpha-1} + \beta B (C/6N)^{-\beta-1} \cdot (-C/6N^2) = 0$:

$$\alpha A N^{-\alpha-1} = \frac{\beta B}{N}\left(\frac{C}{6N}\right)^{-\beta} \Rightarrow \frac{\alpha A}{N^\alpha} = \frac{\beta B}{D^\beta} \tag{11}$$

So at the optimum, the parameter and data terms contribute equally to the reducible loss. Solving jointly with $C = 6ND$ gives the result above. □

For the Chinchilla estimates $\alpha \approx \beta \approx 0.5$: $N^* \propto D^* \propto \sqrt{C}$—equal scaling of parameters and tokens. This overturns the Kaplan (2020) prescription (scale N faster than D) and motivates training smaller models on more data.

Emergent Capabilities as Phase Transitions

Wei et al. (2022) define emergent capabilities as abilities that are "absent in smaller models and present in larger models" with a discontinuous transition. Schaeffer et al. (2023) argue that many apparent discontinuities are artifacts of nonlinear metrics (accuracy) applied to smooth underlying improvements. Under linear metrics, the same models show smooth improvement.

A genuine phase transition requires that the capability is discontinuous in a smooth metric as $N \to \infty$ at fixed C/N. Theoretical results suggest this occurs for tasks requiring k conditions to be simultaneously satisfied: the joint success probability transitions from $O(\varepsilon^k)$ to $O(1)$ as each condition is independently crossed, yielding sharp threshold effects even in smooth metrics when $k \geq 2$.

Cross-Reference to Other Parts. The phase transition perspective connects directly to the spin glass analysis of Chapter 1: the storage capacity transition in Hopfield networks is a sharp first-order transition in the thermodynamic limit, analogous to the emergent capability transition. The Marchenko–Pastur spectral transition at the interpolation threshold $P = N$ is the random matrix analogue of the jamming transition. All three—spin glass capacity, double descent, and emergent capabilities—are manifestations of the same math-

ematical phenomenon: the transition between an underconstrained and overconstrained system as the ratio of parameters to data crosses a critical threshold.

Large-Scale Fine-Tuning

Historical and Motivational Context. *The dominant paradigm in NLP shifted from training task-specific models to fine-tuning large pre-trained models (BERT, 2018; GPT-2, 2019) on small task-specific datasets. The success of this paradigm raised a theoretical question: why does fine-tuning a 100M-parameter model on 10K examples not catastrophically overfit? The answer, provided by the NTK perspective (Part III) and the flat minima interpretation, is that fine-tuning moves the model along directions of low curvature in the loss landscape—staying near the pre-trained model while adapting to the task. LoRA (Hu et al., 2022) formalised this intuition: if fine-tuning moves the model along low-rank directions in weight space, the fine-tuned model can be represented with far fewer parameters.*

LoRA: Mathematical Foundations and Rank Selection

LoRA (Hu et al., 2022) parameterises weight updates as $\Delta W = BA$ with $B \in \mathbb{R}^{m \times r}$ and $A \in \mathbb{R}^{r \times n}$, $r \ll \min(m, n)$.

The SVD of the optimal full update $\Delta W^* = U \Sigma V^T$ shows that the best rank-r approximation (by the Eckart–Young theorem from Part I) achieves Frobenius norm error:

$$\|\Delta W^* - B^* A^*\|_F = \sqrt{\sigma_{r+1}^2 + \cdots + \sigma_R^2} \tag{12}$$

where σ_i are the singular values of ΔW^*. Empirically, the singular value spectrum of fine-tuning updates decays rapidly: the top $r = 8$ singular values capture $> 95\%$ of the total Frobenius norm for most NLP tasks, justifying low-rank fine-tuning.

The LoRA training procedure initialises $B = 0$ (so $\Delta W = 0$ at the start of fine-tuning), and A with Kaiming initialisation. The learning rate for A is scaled by α / r (where α is a hyperparameter) to control the magnitude of the initialised update.

QLoRA (Dettmers et al., 2023) stores frozen weights in NF4 (NormalFloat 4-bit): the data type is chosen to minimise quantisation error for normally distributed weights, using $2^4 = 16$ quantisation levels placed at the quantiles of the standard normal. The LoRA path (BA) remains in full precision, allowing the gradient to correct for quantisation error during training.

ZeRO and Fully Sharded Data Parallelism

Data parallel training (DDP) replicates the full model on each GPU and averages gradients. For a 100B model in mixed precision, this requires $100\text{B} \times 16$ bytes ≈ 1.6 TB per GPU—far exceeding GPU memory.

ZeRO (Zero Redundancy Optimizer, Rajbhandari et al., 2020) shards the three memory-intensive components across N GPUs:

- **ZeRO-1**: Shard optimizer states (Adam: 12 bytes/param). Memory reduction: $4\times$.

- **ZeRO-2**: Also shard gradients (2 bytes/param fp16). Memory reduction: $8\times$.
- **ZeRO-3**: Also shard parameters (2 bytes/param fp16). Memory reduction: $N\times$.

Theorem 5.1 (ZeRO-3 Memory Complexity)**.** *For a model with Ψ parameters and Adam optimizer on N GPUs, ZeRO-3 reduces per-GPU memory from 16Ψ bytes (DDP) to $16\Psi/N +$ buffer bytes. The communication cost is $\sim 3\Psi$ bytes per training step (one allgather for parameters + one reduce-scatter for gradients), comparable to DDP's allreduce cost of 2Ψ bytes.*

Gradient Checkpointing and the Memory-Compute Tradeoff

Full backpropagation requires storing all intermediate activations for use in the backward pass, requiring $O(L \cdot B \cdot d)$ memory for L layers, batch size B, and hidden dimension d.

Theorem 5.2 (Checkpointing Memory-Compute Tradeoff)**.** *With gradient checkpointing at every $\sqrt{L}$ layers, total memory is $O(\sqrt{L} \cdot B \cdot d)$—a $\sqrt{L}$-fold reduction. The compute cost increases by a factor of $\approx 1+1/\sqrt{L}$. Checkpointing at every k layers costs $O(k \cdot B \cdot d + (L/k) \cdot B \cdot d)$ memory, minimised at $k = \sqrt{L}$.*

Cross-Reference to Other Parts. The LoRA low-rank structure is mathematically equivalent to the intrinsic dimensionality hypothesis of representation learning: fine-tuning explores a low-dimensional subspace of weight space, analogous to the low-dimensional data manifold hypothesis of Part I. The NTK analysis of Part III shows that, near the pre-trained model, the loss landscape is approximately quadratic with the NTK as the Hessian—and the low-rank structure of LoRA exploits the fact that the NTK has a small number of large eigenvalues corresponding to the task-relevant directions.

Multimodal Models: Geometry of Joint Representation Spaces

Historical and Motivational Context. *Multimodal learning—training models that understand multiple modalities simultaneously—was attempted before the deep learning era with canonical correlation analysis (CCA, Hotelling 1936) and multi-view learning. The modern era began with CLIP (Radford et al., 2021), which demonstrated that a simple contrastive objective on 400 million internet image-text pairs could produce representations with remarkable zero-shot transfer. The key insight was scale: previous multimodal models were trained on small, curated datasets; CLIP used the internet as a data source, learning from the natural co-occurrence of images and their descriptions.*

CLIP and Contrastive Cross-Modal Alignment

CLIP (Radford et al., 2021) trains an image encoder $f : \mathcal{X} \to \mathbb{R}^d$ and text encoder $g : \mathcal{T} \to \mathbb{R}^d$ on N matched pairs $\{(x_i, t_i)\}$ using a symmetric InfoNCE loss:

$$\mathcal{L}_{\text{CLIP}} = -\frac{1}{2N} \sum_i \left[\log \frac{e^{f(x_i) \cdot g(t_i)/\tau}}{\sum_j e^{f(x_i) \cdot g(t_j)/\tau}} + \log \frac{e^{f(x_i) \cdot g(t_i)/\tau}}{\sum_j e^{f(x_j) \cdot g(t_i)/\tau}} \right] \tag{13}$$

At convergence, the optimal encoders satisfy: $f(x) \cdot g(t) \propto \log p(x,t)/(p(x)p(t))$—representations encode the pointwise mutual information between matched pairs. The temperature τ

controls the sharpness: small τ focuses on hard negatives, large τ provides a softer objective.

Optimal Transport for Cross-Modal Alignment

The CLIP loss treats each pair independently. Optimal transport provides geometric alignment of entire distributions. The Wasserstein distance between image and text embedding distributions:

$$W_p(\mu_f, \mu_g) = \left(\inf_{\gamma \in \Pi(\mu_f, \mu_g)} \int \|z_f - z_g\|^p \, d\gamma(z_f, z_g) \right)^{1/p} \tag{14}$$

The Sinkhorn algorithm computes the regularised OT distance $\mathrm{OT}_\varepsilon(\mu_f, \mu_g) = \min_{\gamma \in \Pi} \langle C, \gamma \rangle + \varepsilon H(\gamma)$ via matrix scaling in $O(n^2/\varepsilon)$ time. OT-based alignment losses produce representations where the embedding distributions are geometrically aligned (not just individual pairs), improving zero-shot transfer.

Flamingo and Cross-Attention Conditioning

Flamingo (Alayrac et al., 2022) interleaves a frozen vision encoder with a frozen language model connected by Perceiver Resampler and cross-attention layers. The Perceiver Resampler compresses variable-length visual features ($T_{\text{vis}} \times d_{\text{vis}}$) to a fixed number of visual tokens ($N_{\text{vis}} \times d_{\text{lm}}$) via cross-attention: $z = \text{CrossAttn}(Q, K = V = \text{visual features})$, $Q \in \mathbb{R}^{N_{\text{vis}} \times d_{\text{lm}}}$.

The cross-attention layers in the LM implement visual conditioning: language tokens attend to visual tokens at every layer, enabling visual information to modulate language generation at every level of the hierarchy.

Unified Multimodal Architectures and Tokenisation

The trend toward unified multimodal architectures (LLaVA, Gemini, GPT-4V) treats all modalities as token sequences in a single autoregressive transformer. Images are tokenised via VQ-VAE discrete codes (enabling generation) or CLIP patch features (enabling understanding); audio via EnCodec tokens; video via spatiotemporal patches. The unified token space is $\mathcal{V} = \mathcal{V}_{\text{text}} \cup \mathcal{V}_{\text{image}} \cup \mathcal{V}_{\text{audio}} \cup \cdots$ with a single softmax over all modalities.

Cross-Reference to Other Parts. The cross-modal contrastive alignment of CLIP is an application of the InfoNCE objective studied for self-supervised learning in Part IV (Chapter 10). The optimal transport alignment connects to the Benamou–Brenier formulation of flow matching (Part IV, Chapter 7): both minimise the W_2 distance between distributions, and both produce straight transport trajectories when the source and target distributions are well-matched. The VQ-VAE tokenisation of images for unified architectures connects the discrete latent spaces of Part IV (Chapter 8) to the token vocabularies of the language models studied in Part II.

Random Matrix Theory and the Neural Tangent Kernel

Historical and Motivational Context. *The Neural Tangent Kernel (NTK) was introduced by Jacot, Gabriel, and Hongler (2018) to analyse the training dynamics of infinitely wide neural networks. The key observation: as the width $n \to \infty$, the network function $f_\theta(x)$ is well approximated by its first-order Taylor expansion around initialisation, and the kernel matrix $K(x, x') = \nabla_\theta f(x) \cdot \nabla_\theta f(x')$ remains constant throughout training (the "frozen kernel" regime). This converts the non-linear dynamics of neural network training into linear kernel regression—a tractable problem with a complete theory.*

The NTK: Definition and Constant-Kernel Theorem

For a neural network $f_\theta : \mathbb{R}^d \to \mathbb{R}$ with parameters $\theta \in \mathbb{R}^P$, the Neural Tangent Kernel is:

$$K_\theta(x, x') = \langle \nabla_\theta f_\theta(x), \nabla_\theta f_\theta(x') \rangle = \sum_p \frac{\partial f_\theta(x)}{\partial \theta_p} \frac{\partial f_\theta(x')}{\partial \theta_p} \tag{15}$$

Theorem 7.1 (NTK Constancy, Jacot et al. 2018)**.** *For a fully-connected network with i.i.d. Gaussian initialisation and activation function σ with bounded second derivative, as the width $n \to \infty$: (i) $f_\theta(x) \to \mathcal{GP}(0, K_\infty)$ at initialisation; (ii) $K_\theta(x, x') \to K_\infty(x, x')$ almost surely at initialisation; (iii) $K_\theta(x, x')$ remains equal to $K_\infty(x, x')$ throughout gradient flow training. Consequently, the training dynamics converge to kernel gradient descent on K_∞.*

Implication for generalisation. Under gradient flow with the MSE loss $\mathcal{L} = \frac{1}{2}\|f_\theta(X) - y\|^2$:

$$\frac{df(X)}{dt} = -K_{XX}\,(f(X) - y), \quad f(X, t) = (I - e^{-K_{XX}t})y \tag{16}$$

where $K_{XX} = K_\infty(X, X) \in \mathbb{R}^{n \times n}$ is the kernel matrix on training data. The solution converges to the kernel interpolant $\hat{f}(x) = k(x, X)K_{XX}^{-1}y$, which is the same as kernel regression (minimum RKHS norm interpolation).

Spectral Bias and the Low-Frequency Preference

The NTK eigenvalue decomposition reveals a key property: neural networks learn low-frequency components of the target function first.

Let $K_{XX} = U\Lambda U^T$ with $\lambda_1 \geq \ldots \geq \lambda_n > 0$. The error in the direction of eigenvector u_i decays as $e^{-\lambda_i t}$: fast for large eigenvalues (large λ_i = high-frequency components of the kernel, corresponding to low-frequency functions), slow for small eigenvalues (low-frequency components of the kernel, corresponding to high-frequency functions). This is the *spectral bias* (Rahaman et al., 2019): neural networks fit low-frequency patterns first.

Beyond the NTK: Feature Learning and the Mean-Field Limit

The NTK regime is a linearisation that does not capture the feature learning that makes deep networks powerful. In the NTK limit, the network's features (first-layer representations) are frozen at their random initialisations; only the last-layer weights effectively change.

The *mean-field* limit (Yang and Hu, 2021; Bordelon and Pehlevan, 2022) studies intermediate regimes where feature learning occurs. The key distinction is the μP parameterisation (Maximal Update Parameterisation): by scaling layer widths differently, the network can remain in a regime where both feature learning and stable training occur.

Cross-Reference to Other Parts. The NTK eigenvalue spectrum is determined by the Marchenko-Pastur distribution discussed in Chapter 1: for random initialisations, the NTK matrix has eigenvalues distributed according to the Marchenko-Pastur law with ratio $\gamma = n/d$ (network width / input dimension). The spectral bias is directly related to the eigenvalue spectrum: features corresponding to small eigenvalues of the NTK matrix are learned slowly, analogous to flat directions in the loss landscape (Chapter 1) which correspond to zero eigenvalues of the Hessian.

Double Descent and the Interpolation Threshold

Historical and Motivational Context. *The classical statistical wisdom, codified in the bias-variance tradeoff and Occam's razor, held that models should be neither too simple (underfitting) nor too complex (overfitting): test error follows a U-shaped curve as model capacity increases. The double descent phenomenon (Belkin et al., 2019; Nakkiran et al., 2020) overturned this: test error follows a double-U curve, with a second descent into low error as model capacity increases into the overparameterised regime. The peak of the second U, called the interpolation threshold, occurs when the model just barely fits the training data and is the worst possible point to deploy a model.*

The Bias-Variance Decomposition and Double Descent

For a predictor $\hat{f}$ trained on n samples from a distribution P:

$$\mathbb{E}[(y - \hat{f}(x))^2] = \underbrace{\text{Bias}^2[\hat{f}(x)]}_{\text{systematic error}} + \underbrace{\text{Var}[\hat{f}(x)]}_{\text{noise from training data}} + \sigma^2 \tag{17}$$

Classical theory: increasing model capacity decreases bias but increases variance; the optimum is at intermediate capacity. Double descent: in the overparameterised regime, both bias *and* variance can decrease simultaneously, as the model finds solutions via implicit regularisation.

Theorem 8.1 (Double Descent for Least Squares, Hastie et al. 2019)**.** *For a linear model $\hat{\beta} = X^+ y$ (pseudoinverse) in the regime $p/n \to \gamma > 1$ (overparameterised), the test risk under the Marchenko-Pastur distribution of features is:*

$$R(\gamma) = \sigma^2 \frac{\gamma}{\gamma - 1} + \|\beta^*\|^2 R_{bias}(\gamma) \tag{18}$$

where $R_{bias}(\gamma) \to 0$ as $\gamma \to \infty$. The risk diverges as $\gamma \to 1^+$ (the interpolation threshold) and decreases to the Bayes risk as $\gamma \to \infty$.

The implicit regularisation mechanism: when $p > n$, the pseudoinverse solution $\hat{\beta} = X^+ y$ is the minimum ℓ_2-norm interpolating solution. This minimum-norm property acts as an implicit regulariser, preferring solutions close to zero—which, under standard signal assumptions, have small generalisation error.

Epoch-Wise Double Descent

Double descent also occurs along the training epoch axis: for a fixed overparameterised model, test error first decreases (underfitting regime), then increases (as the model memorises training data), then decreases again (as continued training finds a better-regularised solution). This epoch-wise double descent explains why early stopping may not be the globally optimal training strategy—continued training beyond the naive early-stopping point can improve generalisation.

Optimisation Dynamics: Loss Landscapes and Flatness

Historical and Motivational Context. *The empirical observation that SGD finds flatter minima than deterministic gradient descent—and that flatter minima generalise better—was documented by Hochreiter and Schmidhuber (1997) and formalised by Keskar et al. (2017), who showed that large-batch training finds sharper minima with worse generalisation. The theoretical explanation via PAC-Bayes (Dziugaite and Roy, 2017) provided the first rigorous connection between sharpness and generalisation: the PAC-Bayes bound with a Gaussian posterior centered at θ has magnitude determined by the KL divergence between the posterior and a fixed prior, which is related to the Hessian eigenvalues (sharpness).*

Sharpness-Aware Minimisation

SAM (Sharpness-Aware Minimisation, Foret et al., 2021) minimises the worst-case loss within a neighbourhood of the current parameters:

$$\min_{\theta} \max_{\|\varepsilon\|_2 \leq \rho} \mathcal{L}(\theta + \varepsilon) \tag{19}$$

The inner maximisation is solved approximately: the worst perturbation is $\varepsilon^* \approx \rho \, \nabla_\theta \mathcal{L} / \|\nabla_\theta \mathcal{L}\|$ (the gradient direction scaled to have norm ρ). SAM then computes the gradient at $\theta + \varepsilon^*$ and takes a gradient step: $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta + \varepsilon^*)$.

The PAC-Bayes–Sharpness Connection

The PAC-Bayes bound with a Gaussian posterior $Q = \mathcal{N}(\theta, \sigma^2 I)$ and standard prior $P = \mathcal{N}(0, \tau^2 I)$:

$$\mathbb{E}_{w \sim Q}[\mathcal{L}(w)] \leq \hat{\mathcal{L}}(\theta) + \frac{1}{n-1}\left[\frac{\|\theta\|^2}{2\tau^2} + \frac{d}{2}\log\frac{\tau^2}{\sigma^2} - \frac{d}{2}\right] + O(1/n) \tag{20}$$

The term $d/(2)\log(\tau^2/\sigma^2)$ penalises sharp minima (σ small means the posterior is concentrated, but then random $w \sim Q$ can be far from θ if the Hessian eigenvalues are large). Flat minima (large σ supported) have smaller KL divergence and tighter PAC-Bayes bounds.

Cross-Reference to Other Parts. The PAC-Bayes sharpness connection links to the information-geometric perspective of Part I: the Fisher information matrix at the optimum measures the curvature of the log-likelihood surface, and flat minima correspond to small Fisher eigenvalues. The SAM objective is a minimax problem analogous to the GAN objective of Part IV: the inner maximisation finds the worst perturbation, and the outer minimisation finds pa-

rameters robust to it.

Graph Neural Networks: Spectral Theory and Expressivity

Historical and Motivational Context. *Graph convolutional networks (Kipf and Welling, 2017) popularised GNNs for node classification, but their theoretical foundations were developed later. The connection between GNNs and the Weisfeiler–Lehman (WL) graph isomorphism test (Xu et al., 2019) revealed the fundamental expressivity limit of message-passing GNNs: they cannot distinguish graphs that the 1-WL test cannot distinguish. This motivated higher-order GNNs (using k-tuples of nodes for k-WL tests) and subgraph GNNs, at the cost of exponentially higher computational complexity.*

Spectral Graph Convolutions

The graph Laplacian $L = D - A$ (where D is the degree matrix and A the adjacency matrix) has eigenvectors $U = [u_1, \dots, u_n]$ and eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$. Graph convolution in the spectral domain:

$$g *_G x = U\, \hat{g}(\Lambda)\, U^T x \tag{21}$$

where $\hat{g}(\Lambda)$ is a learnable spectral filter. ChebNet (Defferrard et al., 2016) approximates $\hat{g}(\lambda) \approx \sum_{k=0}^{K} c_k T_k(\tilde{\lambda})$ via Chebyshev polynomials, avoiding eigendecomposition and achieving $O(K|E|)$ complexity per layer. GCN (Kipf and Welling, 2017) simplifies to first-order Chebyshev: $H' = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}HW)$, where $\tilde{A} = A + I$ (self-loops added).

The Weisfeiler-Lehman Expressivity Bound

Theorem 10.1 (GNN-WL Equivalence, Xu et al. 2019)**.** *Any GNN with injective aggregation and combination functions is at least as powerful as the 1-WL test. Any 1-WL-indistinguishable pair of graphs is also GNN-indistinguishable for any MPNN. GIN (Graph Isomorphism Network) achieves maximum 1-WL expressivity by using a sum aggregator with a learnable ε:*

$$h_v^{(k)} = \text{MLP}^{(k)}\left((1 + \varepsilon^{(k)})h_v^{(k-1)} + \sum_{u \in \mathcal{N}(v)} h_u^{(k-1)}\right) \tag{22}$$

Sum aggregation is strictly more powerful than mean or max: it distinguishes graphs with different counts of a substructure, while mean is insensitive to node counts and max ignores multiplicity.

Mixture of Experts and Sparse Architectures

Historical and Motivational Context. *Mixture of Experts (MoE) was introduced by Jacobs et al. (1991) as a way to divide a complex problem into subproblems, each solved by a specialised expert network. The gating function routes each input to the most appropriate expert. The method was extended to hierarchical MoE and applied to language modelling by Shazeer et al. (2017), who introduced Sparsely-Gated MoE: only the top-k experts (typically $k = 1$ or $k = 2$) process each*

input token, while the rest remain inactive. This achieves the same model capacity as a dense model with E times the parameters at a fraction of the compute: a 500B-parameter sparse model with $k = 2$ out of $E = 64$ experts uses approximately $2/64 = 3\%$ of its parameters per token, achieving dense-equivalent quality at much lower FLOPs per token.

Sparsely-Gated MoE: Mathematical Formulation

An MoE layer replaces a standard FFN with E expert networks $\{f_1, \ldots, f_E\}$ and a gating network $g(x) \in \mathbb{R}^E$. The output:

$$\text{MoE}(x) = \sum_{i=1}^{E} g_i(x)\, f_i(x) \tag{23}$$

For sparse routing, only the top-k gates are non-zero: $g_i(x) = \text{softmax}_{\text{top-}k}(W_g x)_i$ where softmax_top-k sets all but the k largest logits to $-\infty$.

Expert collapse. Without load balancing, routing converges to sending all tokens to the same expert (because a small perturbation making one expert better causes it to receive more tokens, making it better, a positive feedback loop). The auxiliary load-balancing loss prevents this:

$$\mathcal{L}_{\text{aux}} = \alpha E \sum_{i=1}^{E} f_i \cdot P_i \tag{24}$$

where f_i is the fraction of tokens routed to expert i and $P_i = \frac{1}{T}\sum_x \text{softmax}(W_g x)_i$ is the fraction of router probability assigned to expert i. Minimising $\mathcal{L}_{\text{aux}}$ encourages uniform load distribution.

Expert Choice Routing and the Load Balancing Theorem

Expert Choice routing (Zhou et al., 2022) inverts the routing direction: rather than tokens choosing experts, experts choose tokens. Each expert selects the top-k tokens with the highest gating scores. This guarantees perfect load balance by construction (each expert always processes exactly k tokens) but means some tokens may be processed by more than one expert and others by none.

Theorem 11.1 (Expert Choice Load Balance)**.** *With Expert Choice routing where each of E experts selects top-k tokens from a batch of T tokens, each expert processes exactly k tokens. The total computation is Ek token-expert interactions, compared to Tk for standard top-k token routing. For $k = T/E$ (matching the expected load), both use the same total compute, but Expert Choice guarantees equal load while token routing has load variance $O(T/E)$.*

DeepSeek MoE and Expert Specialisation

DeepSeekMoE (Dai et al., 2024) introduces fine-grained expert segmentation: instead of E large experts, use mE small experts of $1/m$ the size each, selecting mk of them per token. With $m = 4$: quadruple the expert count, reduce each expert to $1/4$ size, select $4\times$ more experts per token. This allows finer-grained specialisation: rather than a few coarse experts ("one for code, one for math, one for language"), fine-grained

experts develop micro-specialisations. Empirically, fine-grained MoE with the same total compute outperforms coarse MoE and approaches dense model performance.

Cross-Reference to Other Parts. The MoE architecture connects to the mechanistic interpretability of Part V (Chapter 7): individual experts develop specialised functions that can be probed, analogous to the modular circuits identified in transformer attention heads. The load balancing theorem is a combinatorial optimisation result related to the optimal transport theory of Part IV: routing tokens to experts is a bipartite matching problem, and the OT solution (Sinkhorn routing) provides an optimal load-balanced assignment.

Knowledge Distillation and Model Compression

Historical and Motivational Context. *Knowledge distillation (Hinton, Vinyals, and Dean, 2015) was introduced as a method for transferring knowledge from a large "teacher" model to a small "student" model, allowing the student to match the teacher's performance at a fraction of the compute. The key insight: the teacher's output probabilities ("soft targets") contain more information than the hard labels—a near-zero probability for the wrong class still encodes information about the relative similarity of classes. This "dark knowledge" is accessible to the student when it is trained to match the teacher's soft predictions.*

The Distillation Objective and Temperature Scaling

The standard distillation loss combines a soft loss (matching teacher's probabilities) with a hard loss (matching ground-truth labels):

$$\mathcal{L}_{\text{KD}} = (1-\alpha)\mathcal{L}_{\text{CE}}(y, \sigma(z_S)) + \alpha T^2 \mathcal{L}_{\text{CE}}(\sigma(z_T/T), \sigma(z_S/T)) \tag{25}$$

where z_S and z_T are student and teacher logits, $T > 1$ is the temperature, and α balances the two objectives. The factor T^2 compensates for the reduced gradient magnitude of the soft targets (gradients scale as $1/T^2$).

Why high temperature works. At temperature T, the soft targets $\sigma(z_T/T)$ distribute probability mass more evenly. For a class with logit $z = -10$, at $T = 1$: $\sigma(-10) \approx 5 \times 10^{-5}$ (near zero); at $T = 10$: $\sigma(-1) \approx 0.27$ (significant). The high-temperature distribution reveals the relative similarities between classes that are invisible in the $T = 1$ output.

Feature-Level and Relation-Level Distillation

Intermediate feature distillation (Romero et al., 2015) trains the student to match the teacher's intermediate representations, not just the final output: $\mathcal{L}_{\text{hint}} = \|f_S(x) - W_r\, f_T(x)\|^2$ where W_r is a learned regression matrix mapping teacher features to the student's dimensionality. Relation-based distillation (Park et al., 2019) matches the pairwise relationship structure: $\mathcal{L}_{\text{RKD}} = \sum_{(i,j)} \ell(d_S(x_i, x_j), d_T(x_i, x_j))$ where $d(x_i, x_j)$ is a pairwise distance in feature space.

Speculative Decoding and Draft-Verify Architecture

Speculative decoding (Leviathan et al., 2023; Chen et al., 2023) accelerates inference by using a small draft model to generate k tokens, then verifying all k tokens in parallel with the large target model:

1. Draft model M_q autoregressively generates $x_1, \ldots, x_k$.
2. Target model M_p computes $p(x_{t+1}|x_1, \ldots, x_t)$ for all t in parallel.
3. Accept token x_t if $u \leq \min(1, p(x_t)/q(x_t))$ for $u \sim \text{Uniform}(0, 1)$.

This acceptance rule ensures that the joint distribution of accepted tokens matches the target model M_p—speculative decoding produces identical output distributions to the target model while running $k/\bar{\alpha}$ times faster, where $\bar{\alpha}$ is the acceptance rate.

Theorem 12.1 (Speculative Decoding Unbiasedness)**.** *The distribution over accepted sequences under the draft-verify algorithm with acceptance probability* $\min(1, p(x)/q(x))$ *is identical to the target distribution* p*. Proof: at each position, the marginal probability of accepting token* x *is* $q(x) \cdot \min(1, p(x)/q(x)) = \min(q(x), p(x))$*. After rejection, a fresh sample from* p *is drawn. The mixture* $P(\text{accept} = x) + P(\text{reject}) \cdot p'(x) = p(x)$ *for any token* x*, where* p' *is the adjusted distribution.* □

Attention Variants and Efficient Transformers

Multi-Query Attention and Grouped-Query Attention

Standard multi-head attention (MHA) uses H heads, each with separate Q_h, K_h, V_h projections. The KV cache stores all K_h, V_h for all previous tokens, using $2HLd$ memory for sequence length L.

Multi-Query Attention (MQA, Shazeer, 2019) shares a single K, V across all heads: H separate Q_h projections but shared K, V. KV cache memory: $2Ld$ ($H\times$ reduction). Grouped-Query Attention (GQA, Ainslie et al., 2023) uses G groups: H/G heads share one KV pair. GQA with $G = H$ is MHA; $G = 1$ is MQA. GQA at $G = 8$ achieves near-MHA quality with near-MQA memory, and is the default in modern large language models (Llama 2/3, Gemma, Mistral).

Sliding Window Attention and Longformer

For long sequences, full attention ($O(L^2)$) is prohibitive. Sliding window attention (Longformer, Beltagy et al., 2020) restricts each token to attend only to its w-neighbourhood: $a_{ij} = q_i^T k_j/\sqrt{d}$ for $|i - j| \leq w/2$, else $-\infty$. This gives $O(Lw)$ attention, reducing from $O(L^2)$ to $O(Lw)$ for window $w \ll L$.

Combined with global attention (specific tokens attend globally, e.g. CLS, question tokens), the Longformer captures both local context and global structure. The theoretical motivation: natural text has a hierarchical structure—local attention captures phrase-level semantics; global attention captures discourse.

Cross-Reference to Other Parts. The sliding window attention is the temporal analogue of the spatial local attention in CNNs (Part II): both restrict each position's receptive field to a local neighbourhood, motivated by the locality of the relevant structure. The GQA memory reduction complements the ZeRO-3 parameter sharding (Chapter 5 of this part): ZeRO-3 reduces model state memory; GQA reduces KV cache memory during inference—the two bottlenecks are complementary and both are addressed in modern deployment.

Bayesian Deep Learning and Uncertainty Quantification

Historical and Motivational Context. *Bayesian neural networks—treating network weights as probability distributions rather than point estimates—were proposed by MacKay (1992) and Neal (1996). Neal showed that an infinitely wide one-hidden-layer Bayesian neural network with Gaussian priors on weights converges to a Gaussian process, establishing the deep connection between Bayesian NNs and kernel methods that the NTK later formalised. The practical challenge of Bayesian NNs is the intractability of the posterior $p(\theta|\mathcal{D})$—exact inference requires integrating over all possible weight configurations, which is intractable for modern networks. Approximate inference methods (variational Bayes, MCMC, Laplace approximation) provide tractable alternatives with different accuracy-compute tradeoffs.*

Variational Bayes and Mean-Field Approximation

Variational Bayes (VB) approximates the posterior $p(\theta|\mathcal{D})$ with a tractable distribution $q_\phi(\theta)$ by minimising the KL divergence:

$$q^* = \arg\min_{q_\phi} D_{KL}(q_\phi(\theta)\|p(\theta|\mathcal{D})) = \arg\max_{q_\phi} \underbrace{\mathbb{E}_{q_\phi}[\log p(\mathcal{D}|\theta)] - D_{KL}(q_\phi\|p(\theta))}_{\text{ELBO}} \tag{26}$$

The mean-field approximation assumes $q_\phi(\theta) = \prod_i q_\phi(\theta_i)$ (fully factored), enabling efficient computation but missing posterior correlations. For Gaussian mean-field: $q_\phi(\theta_i) = \mathcal{N}(\mu_i, \sigma_i^2)$, the local reparametrisation trick (Kingma et al., 2015) enables gradient computation without variance explosion.

Monte Carlo Dropout as Approximate Bayes

Dropout (Srivastava et al., 2014) randomly zeros out units during training with probability p. Gal and Ghahramani (2016) showed that a network with dropout applied at every weight layer is equivalent to a specific variational Bayes approximation with a Bernoulli approximate posterior. Inference-time dropout (keeping dropout active at test time) thus performs approximate Bayesian inference: running T forward passes produces samples from the approximate posterior predictive distribution:

$$p(y^*|x^*, \mathcal{D}) \approx \frac{1}{T}\sum_{t=1}^{T} p(y^*|x^*, \hat{\theta}_t), \quad \hat{\theta}_t \sim q_\phi(\theta) \tag{27}$$

The predictive uncertainty has two components: *aleatoric uncertainty* (irreducible noise in the data) and *epistemic uncertainty* (uncertainty about the model parameters, reducible with more data). The variance of the T predictions captures epistemic uncertainty; the mean of the per-prediction variances captures aleatoric uncertainty.

The Laplace Approximation

The Laplace approximation fits a Gaussian to the posterior $p(\theta|\mathcal{D})$ at the MAP estimate θ^*:

$$p(\theta|\mathcal{D}) \approx \mathcal{N}(\theta^*,\ \Lambda^{-1}), \quad \Lambda = -\nabla_\theta^2 \log p(\theta|\mathcal{D})\big|_{\theta^*} \tag{28}$$

where Λ is the Hessian (negative of the log-posterior curvature). The predictive distribution is: $p(y^*|x^*, \mathcal{D}) \approx \int p(y^*|x^*, \theta)\mathcal{N}(\theta; \theta^*, \Lambda^{-1})d\theta$.

For neural networks, Λ is too large to store (P^2 for P parameters). The Kronecker-factored Laplace approximation (Ritter et al., 2018) uses the KFAC approximation of Λ from Part I: $\Lambda \approx \bigotimes_\ell A_\ell \otimes G_\ell$, enabling scalable Bayesian inference at the cost of diagonal-block accuracy.

Cross-Reference to Other Parts. The Laplace approximation is the second-order Taylor expansion of the log-posterior around the MAP—the Bayesian analogue of the K-FAC approximation to the Fisher information matrix developed in Part I. Both exploit the Kronecker-factored structure of the Hessian to enable tractable second-order computation. The ELBO objective of variational Bayes is the Bayesian analogue of the VAE objective of Part IV: both optimise a lower bound on the marginal likelihood by introducing an approximate posterior and minimising KL divergence.

Conformal Prediction and Distribution-Free Uncertainty

Historical and Motivational Context. *Conformal prediction (Vovk, Gammerman, and Shafer, 1999) provides distribution-free prediction sets with guaranteed finite-sample coverage. Unlike Bayesian methods (which require a prior) or frequentist methods (which require distributional assumptions), conformal prediction guarantees $P(y^* \in \hat{C}(x^*)) \geq 1 - \alpha$ for any significance level α, for any distribution $P(x, y)$, based only on the exchangeability of the data. The approach was popularised for deep learning by Angelopoulos and Bates (2021), who showed that conformal prediction can be applied on top of any pre-trained model without retraining.*

Split Conformal Prediction

Split conformal prediction (Papadopoulos et al., 2002) uses a held-out calibration set $\{(x_i, y_i)\}_{i=1}^n$ to compute nonconformity scores $s_i = s(x_i, y_i)$, then at test time:

$$\hat{C}(x^*) = \{y : s(x^*, y) \leq \hat{q}^+_{1-\alpha}\} \tag{29}$$

where $\hat{q}^+_{1-\alpha}$ is the $\lceil (1-\alpha)(n+1) \rceil / n$-quantile of $\{s_1, \ldots, s_n\}$.

Theorem 15.1 (Conformal Coverage Guarantee)**.** *For any distribution P over (x, y) and any exchangeable calibration set, split conformal prediction with nonconformity score s achieves: $P(y^* \in \hat{C}(x^*)) \geq 1 - \alpha$. Under the additional assumption of distinct nonconformity scores: $P(y^* \in \hat{C}(x^*)) \leq 1 - \alpha + 1/(n+1)$, so the coverage is exactly $1 - \alpha$ up to a $1/(n+1)$ slack.*

Application to classification. The nonconformity score for a softmax classifier is $s(x, y) = 1 - f(x)_y$ (one minus the predicted probability of the true class). The conformal prediction set $\hat{C}(x^*)$ contains all classes y with predicted probability above the threshold $1 - \hat{q}$.

Risk-Controlling Prediction Sets

Risk-Controlling Prediction Sets (RCPS, Bates et al., 2021) generalise conformal prediction to arbitrary risk measures beyond marginal coverage: find the smallest prediction set size λ^* such that $\mathbb{E}[L(\hat{C}_\lambda(x^*), y^*)] \leq \alpha$, where L is any monotone risk functional (e.g. false negative rate, expected set size, F1 score). This enables calibration of prediction

sets for high-stakes applications like medical diagnosis, where false negatives (missing the true class) are far more costly than false positives.

Federated Learning and Privacy-Preserving ML

Historical and Motivational Context. *Federated learning (McMahan et al., 2017) was introduced by Google to train models on mobile devices without centralising sensitive data. The key constraint: raw data never leaves the device; only model updates are shared with a central server. This raised two mathematical problems: (1) communication efficiency— gradients are large and communication is expensive; (2) privacy—even model updates may leak information about individual training examples. The former is addressed by federated averaging and gradient compression; the latter by differential privacy.*

Federated Averaging and Non-IID Data

FedAvg (McMahan et al., 2017) trains a global model by having each client k perform E local gradient descent steps on its local data $\mathcal{D}_k$, then averaging the resulting model weights: $\theta^{(t+1)} = \sum_k \frac{n_k}{n} \theta_k^{(t+1)}$ where $n_k = |\mathcal{D}_k|$ and $n = \sum_k n_k$.

The convergence of FedAvg under non-IID data (where different clients have different data distributions) is challenging: client drift occurs when local optima for individual clients diverge from the global optimum. The bound on the convergence gap: $\|\nabla \mathcal{L}(\theta^T)\|^2 \leq O(\frac{\sigma^2}{T} + \frac{\Gamma E^2}{T})$ where σ^2 is the gradient variance and Γ measures data heterogeneity (the divergence between local and global optima). FedProx (Li et al., 2020) adds a proximal term $\frac{\mu}{2}\|\theta_k - \theta^{(t)}\|^2$ to each local objective, limiting client drift.

Differential Privacy: The Gaussian Mechanism

(ε, δ)-Differential Privacy (Dwork et al., 2006) guarantees that the output distribution of an algorithm $\mathcal{M}$ changes by at most a multiplicative factor e^ε (plus additive δ) when one training example is changed: $P(\mathcal{M}(\mathcal{D}) \in S) \leq e^\varepsilon \, P(\mathcal{M}(\mathcal{D}') \in S) + \delta$ for all adjacent datasets $\mathcal{D}, \mathcal{D}'$ differing in one example.

The Gaussian Mechanism achieves (ε, δ)-DP by adding Gaussian noise to the gradient before sharing: $\tilde{g} = \frac{1}{L}\sum_i \text{clip}(g_i, C) + \mathcal{N}(0, \sigma^2 C^2 I)$ where C is the clipping norm and $\sigma \geq \sqrt{2\log(1.25/\delta)}/\varepsilon$. DP-SGD (Abadi et al., 2016) applies this mechanism at each gradient step, with privacy cost accumulated via the moments accountant.

Cross-Reference to Other Parts. Differential privacy connects to the generalisation theory of Part III via the algorithmic stability framework: a DP algorithm with privacy parameter ε is also ε-stable (changing one training example changes the output by at most e^ε), and algorithmic stability implies generalisation bounds of $O(\varepsilon/n)$—stronger than standard ERM bounds for small ε. The federated averaging framework is an application of distributed optimisation (Part VI, Chapter 5 ZeRO/FSDP): both shard computation across machines, but federated learning adds the constraint that raw data never leaves the local machine.

Test-Time Compute and Inference Scaling

Historical and Motivational Context. *The success of OpenAI o1 (2024) demonstrated that* inference-time compute *can improve accuracy on reasoning tasks comparably to training-time scale: spending more FLOPs at test time (via extended chain-of-thought, beam search, or multiple samples) improves performance without any additional training. This discovery shifted the scaling paradigm: rather than asking "how large should the model be?" one now asks "how much inference compute should be spent?"—and the answer depends on the task difficulty.*

Optimal Test-Time Compute Allocation

The test-time compute problem: given a fixed budget B of inference FLOPs, how should they be spent? Options include:

- **Best-of-N:** generate N independent samples, select the best by a verifier. Cost: $O(N)$ generation calls.
- **Beam search:** maintain B beams, expanding the most promising at each step. Cost: $O(B \cdot L)$ tokens where L is output length.
- **Sequential refinement:** generate a draft, critique it, revise, repeat. Cost: $O(K \cdot L)$ tokens for K refinement steps.
- **Tree search (MCTS-style):** explore a tree of partial solutions, guided by a value function. Cost: $O(T \cdot L)$ for T tree nodes.

Theorem 17.1 (Test-Time Compute Scaling Law, Snell et al. 2024)**.** *For a verifiable task (where correct answers can be checked), best-of-N achieves accuracy $A(N) \approx 1 - (1-p)^N$ where p is the single-sample correctness probability. This grows approximately as $A(N) \approx pN$ for small N and saturates at 1 for large N. The compute-optimal strategy switches from best-of-N to sequential refinement when the marginal gain of an additional sample falls below the marginal gain of a refinement step.*

Verifier Training and the Process Reward Connection

Efficient test-time compute requires a verifier $V(x, y) \in [0, 1]$ that scores candidate answers. Training verifiers is harder than training generators: verifiers must generalise to arbitrary outputs, including ones the model has never produced. Two approaches:

Outcome Reward Models (ORMs): train on (problem, solution, correct/wrong) triples. Fast to train but uninformative about *where* solutions go wrong.

Process Reward Models (PRMs): train on (problem, step, correct/wrong) annotations at every reasoning step. Slower to annotate but provides richer supervision enabling early termination of bad solution paths.

The connection to RLHF: PRMs are step-level reward models, and RL with PRM rewards corresponds to RLHF where each reasoning step is an action and the PRM is the reward signal at each step. This gives a tighter credit assignment signal than ORM-based RLHF, addressing one of the fundamental challenges of long-horizon RL.

Numerical Precision, Mixed Precision Training, and FP8

Historical and Motivational Context. *Neural network training traditionally used 32-bit floating point (FP32) with full precision for both forward and backward passes. Micikevicius et al. (2018) showed that mixed-precision training (FP16 forward, FP32 master weights) reduces memory and improves throughput by $2\times$ with no accuracy loss, provided that (1) a loss scaling factor prevents FP16 underflow, and (2) master weights are maintained in FP32. The subsequent introduction of BFloat16 (FP16 with wider exponent range, matching FP32's dynamic range) simplified loss scaling. FP8 (Micikevicius et al., 2022) further halved memory requirements, enabling training of GPT-4-scale models within practical memory budgets.*

Floating Point Representation and Quantisation Error

An n-bit floating point number with e exponent bits and $m = n - 1 - e$ mantissa bits represents: $(-1)^s \cdot 2^{E-\text{bias}} \cdot (1+f)$ where $E \in [0, 2^e - 2]$ and $f = \sum_{i=1}^{m} b_i 2^{-i}$. The relative rounding error (machine epsilon) is $\varepsilon_{\text{mach}} = 2^{-m}$.

For FP16: $m = 10$, $\varepsilon_{\text{mach}} \approx 10^{-3}$; dynamic range $[6 \times 10^{-5}, 65504]$. For BF16: $m = 7$, $\varepsilon_{\text{mach}} \approx 10^{-2}$; dynamic range $[10^{-38}, 10^{38}]$. For FP8 E4M3: $m = 3$, $\varepsilon_{\text{mach}} \approx 0.125$; dynamic range $[-448, 448]$.

Block-wise Quantisation and GPTQ

GPTQ (Frantar et al., 2023) achieves 4-bit quantisation of large language models with negligible perplexity loss by using an optimal brain compression approach. For a weight matrix $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, quantise column j to $\hat{W}_{:,j} = Q(W_{:,j})$ and update remaining columns to compensate for the quantisation error:

$$W_{:,j+1:} \leftarrow W_{:,j+1:} - (W_{:,j} - \hat{W}_{:,j}) \cdot \frac{H_{j,j+1:}}{H_{jj}} \tag{30}$$

where $H = (X^T X)^{-1}$ is the inverse Hessian of the layer's squared error with respect to its inputs X. This greedy column-by-column quantisation with Hessian-weighted error propagation minimises the total output distortion $\|Wx - \hat{W}x\|^2$ measured over the calibration dataset.

Cross-Reference to Other Parts. GPTQ is the inference analogue of LoRA (Chapter 5 this Part): LoRA reduces the number of trainable parameters by low-rank decomposition during fine-tuning, while GPTQ reduces the precision of stored weights by optimal quantisation during inference. Both exploit the low-effective-rank structure of neural network weight matrices (established by the NTK analysis of Chapter 7): the most important directions are preserved (LoRA: top singular values; GPTQ: directions with highest Hessian weight), and the less important directions are compressed.

Concluding Remarks on Part VI and the Treatise

The eight parts of this treatise have traced a complete arc: from the geometric foundations of linear algebra and information geometry, through the mathematical logic of neural network architectures, into statistical and Bayesian frameworks for learning,

the generative paradigm, sequential decision theory of RL and alignment, and finally the advanced structures that provide the deepest understanding of why deep learning works.

Three mathematical themes recur throughout:

Geometry of high-dimensional probability distributions. The Fisher metric, optimal transport, information geometry of exponential families, and the Riemannian structure of the statistical manifold appear in natural gradient descent, WGAN, flow matching, and CLIP alignment alike. Every learning algorithm is, at its core, a method for navigating the geometry of the space of distributions.

Symmetry and equivariance. From the Transformer's permutation equivariance to the SE(3)-equivariance of molecular models, symmetry constraints are the primary mechanism by which architectural inductive biases are encoded mathematically. Group representation theory provides the language for specifying precisely what symmetries a model respects.

Overparameterisation and generalisation. Double descent, the NTK regime, benign overfitting, and the flat minima hypothesis all address the central mystery of why models with more parameters than data can generalise. Statistical physics—jamming transitions, spin glasses, replica method—provides a quantitative theory of this phenomenon, connecting neural network training to the rich mathematics of disordered systems.

Appendix: Key Formulae Reference

A.1 Statistical Physics

Hopfield energy: $E(x) = -\frac{1}{2}x^T W x, \quad W = \frac{1}{n}\sum_\mu \xi^\mu (\xi^\mu)^T$

Marchenko-Pastur support: $\lambda_\pm = \sigma^2(1 \pm \sqrt{\gamma})^2, \quad \gamma = p/n$

Free additive convolution: $R_{A+B}(z) = R_A(z) + R_B(z)$ for free matrices

Double descent risk (linear): $R(\gamma) = \sigma^2 \frac{\gamma}{\gamma - 1} + \|\beta^*\|^2 R_{\text{bias}}(\gamma)$ for $\gamma > 1$

A.2 Scaling Laws

Chinchilla loss decomposition: $L(N, D) = E + AN^{-\alpha} + BD^{-\beta}, \quad \alpha, \beta \approx 0.5$

Compute-optimal scaling: $N^*(C) = (\alpha A / \beta B)^{1/(\alpha+\beta)} C^{\beta/(\alpha+\beta)}, \quad D^*(C) = C/(6N^*)$

A.3 Fine-Tuning

LoRA: $W_{\text{fine}} = W_0 + BA, \quad B \in \mathbb{R}^{m \times r}, A \in \mathbb{R}^{r \times n}, r \ll \min(m, n)$

Optimal LoRA error: $\|\Delta W^* - B^* A^*\|_F = \sqrt{\sigma_{r+1}^2 + \cdots + \sigma_R^2}$

Gradient checkpointing (optimal): Memory $O(\sqrt{L} \cdot B \cdot d)$ with $O(1)$ extra compute

A.4 Multimodal Alignment

CLIP loss: $\mathcal{L}_{\text{CLIP}} = -\frac{1}{2N}\sum_i [\log \frac{e^{f(x_i)\cdot g(t_i)/\tau}}{\sum_j e^{f(x_i)\cdot g(t_j)/\tau}} + \log \frac{e^{f(x_i)\cdot g(t_i)/\tau}}{\sum_j e^{f(x_j)\cdot g(t_i)/\tau}}]$

Optimal CLIP representation: $f(x)\cdot g(t) \propto \log p(x,t)/(p(x)p(t))$ (pointwise mutual information)

A.5 NTK and Generalisation

NTK: $K_\theta(x,x') = \langle \nabla_\theta f_\theta(x), \nabla_\theta f_\theta(x')\rangle$

NTK training dynamics (infinite-width): $f(X,t) = (I - e^{-K_{XX}t})y$

PAC-Bayes bound (Gaussian): $\mathbb{E}_{w\sim Q}[\mathcal{L}(w)] \le \hat{\mathcal{L}}(\theta) + \frac{1}{n-1}[D_{KL}(\mathcal{N}(\theta,\sigma^2 I)\|\mathcal{N}(0,\tau^2 I))] + O(1/n)$

MATHEMATICS OF DEEP LEARNING

PART VII

SCIENTIFIC AI AND GEOMETRIC DEEP LEARNING

SE(3) Equivariance · AlphaFold 2 and 3

ESM-3 · AlphaGenome · Scientific Foundation Models · Equivariant GNNs

Eight-Part Monograph Series on the Mathematical Principles of Neural Networks

Independent of Companion Volumes — Complete in Itself

Silvano Salvador

© Mathematics of Deep Learning Series

PART VII

SCIENTIFIC AI AND GEOMETRIC DEEP LEARNING

SE(3) Equivariance · AlphaFold 2 and 3

ESM-3 · AlphaGenome · Scientific Foundation Models · Equivariant GNNs

© Mathematics of Deep Learning Series

Silvano Salvador

Preface: The Mathematics of Scientific AI

Historical and Motivational Context. *The history of computational protein structure prediction spans over five decades. Anfinsen's Nobel Prize (1972) established that protein structure is determined by sequence—the thermodynamic hypothesis: the native fold is the global minimum of the free energy. This motivated computational approaches: molecular dynamics (simulating every atom's trajectory), homology modelling (copying structure from known homologues), fragment assembly (Rosetta, Baker et al. 2001). All of these either required excessive computation, failed for proteins without known homologues, or relied on heuristics that couldn't be systematically improved. The CASP (Critical Assessment of Protein Structure Prediction) competition, held biannually since 1994, tracked progress: for twenty years, the best methods achieved ∼3-5Å RMSD on medium-difficulty targets—not bad, but not good enough for drug design. AlphaFold2 (Jumper et al., 2021) achieved ∼1Å RMSD, matching experimental accuracy, winning CASP14 by a margin that prompted CASP organisers to declare the problem "largely solved." The key difference from all predecessors: AlphaFold2 combined evolutionary information (multiple sequence alignments), coevolutionary signals, and end-to-end differentiable $SE(3)$-equivariant architecture, trained on the full PDB. The mathematics that makes this possible is the theory of group equivariance—the subject of this part.*

Scientific AI applies deep learning to problems in the natural sciences where the data has rich geometric and physical structure that must be encoded in the architecture. The central mathematical challenge: biological and chemical objects exist in three-dimensional space and obey physical symmetries that must be respected exactly. A protein structure prediction model that is not equivariant to rotations and translations in 3D space makes predictions that are physically meaningless.

Cross-Reference to Other Parts. The group theory developed in this part is the natural continuation of the symmetry arguments in Part I: the Fisher information metric is invariant under reparameterisation (a group action), and the natural gradient is the gradient in the metric induced by this invariance. The equivariant graph networks of Chapter 6 are equivariant versions of the graph neural networks whose expressivity was analysed in Part II (Chapter 3) via the Weisfeiler–Lehman hierarchy. The diffusion module in AlphaFold 3

(Chapter 3) is a direct application of the DDPM/DDIM framework of Part IV (Chapter 5), but operating on 3D coordinates rather than image pixels.

Groups, Symmetries, and Equivariant Maps

Historical and Motivational Context. *The mathematical connection between symmetry and the structure of physical laws was articulated by Emmy Noether (1915): every continuous symmetry of a physical system corresponds to a conserved quantity. Einstein's general relativity is covariant under diffeomorphisms; quantum mechanics is covariant under $U(n)$ for n-particle systems; molecular physics is covariant under $SE(3)$ and the relevant point group. The application of these ideas to deep learning as an architectural principle—rather than merely a physical observation—was developed in the Geometric Deep Learning framework (Bronstein et al., 2021), which unified GNNs, CNNs, transformers, and equivariant networks as special cases of the same blueprint: choose a symmetry group G, choose G-equivariant layers, and stack them.*

Symmetry Groups in Scientific AI: Definitions and Examples

Definition 1.1 (Group Representation)**.** A *representation* of a group G on a vector space V is a group homomorphism $\rho : G \to \mathrm{GL}(V)$, i.e., a map such that $\rho(gh) = \rho(g)\rho(h)$ for all $g, h \in G$. An *equivariant map* $f : V \to W$ between two representation spaces satisfies $f(\rho_V(g)v) = \rho_W(g)f(v)$ for all $g \in G$ and $v \in V$.

The three primary symmetry groups in scientific AI are:

S_n **(Symmetric group):** the permutation group on n elements. Relevant for problems where inputs are sets—molecules as sets of atoms, sequences as sets of amino acids. Attention is S_n-equivariant: permuting input tokens permutes output tokens accordingly.

SE(3): rigid body motions—rotations $\mathrm{SO}(3)$ and translations $\mathbb{R}^3$. A protein has the same energy, function, and fold regardless of how it is oriented or positioned in 3D space. Formally, $\mathrm{SE}(3) = \mathrm{SO}(3) \ltimes \mathbb{R}^3$ (semidirect product): the action of $(R, t) \in \mathrm{SE}(3)$ on a point $x \in \mathbb{R}^3$ is $(R, t) \cdot x = Rx + t$.

SE(3) $\times$ S_n**:** for molecules as 3D point clouds—both the assignment of atoms to positions must be permutation-equivariant and the 3D coordinates must be $\mathrm{SE}(3)$-equivariant.

Theorem 1.2 (Schur's Lemma and Equivariant Linear Layers)**.** *Let $\rho_i : G \to \mathrm{GL}(V_i)$ and $\rho_j : G \to \mathrm{GL}(V_j)$ be two irreducible representations of a group G. A linear map $L : V_i \to V_j$ is G-equivariant if and only if:*

1. *$V_i \cong V_j$ (the representations are isomorphic), in which case $L = \lambda I$ for some scalar λ.*
2. *$V_i \ncong V_j$, in which case $L = 0$.*

An equivariant linear layer between general representations (direct sums of irreps) is block-diagonal with each block being a scalar multiple of the identity on matching irreps—far fewer parameters than a general linear layer.

Consequence for architecture design. An equivariant layer mapping $V = \bigoplus_\ell n_\ell V^\ell$ to $W =$

$\bigoplus_{\ell'} m_{\ell'} V^{\ell'}$ has at most $\sum_\ell n_\ell m_\ell$ free parameters (one scalar per matching irrep pair), versus $(\sum_\ell (2\ell+1) n_\ell) \times (\sum_{\ell'} (2\ell'+1) m_{\ell'})$ for a general linear layer. Equivariance is a strict architectural constraint that reduces the parameter count while encoding the correct symmetry.

Irreps of SO(3) and Spherical Harmonics: Full Construction

The irreducible representations of $\mathrm{SO}(3)$ are indexed by $\ell \in \mathbb{N}_0$ and have dimension $2\ell+1$. The basis functions are the real spherical harmonics $\{Y_\ell^m\}_{m=-\ell}^{\ell}$, functions on the sphere S^2 that transform under rotation $R \in \mathrm{SO}(3)$ as:

$$(R \cdot Y_\ell^m)(\hat{r}) = Y_\ell^m(R^{-1}\hat{r}) = \sum_{m'} D^{(\ell)}_{m'm}(R)\, Y_\ell^{m'}(\hat{r}) \tag{1}$$

where $D^{(\ell)}(R)$ is the Wigner D-matrix. Key types:

- $\ell = 0$: scalar, invariant under rotation (1-dimensional)
- $\ell = 1$: vector, transforms as a 3D vector (3-dimensional)
- $\ell = 2$: rank-2 symmetric traceless tensor (5-dimensional)
- $\ell = 3$: rank-3 symmetric traceless tensor (7-dimensional)

Explicit construction of Y_ℓ^m for $\ell = 1$. In Cartesian coordinates, the three $\ell = 1$ harmonics are: $Y_1^{-1}(\hat{r}) = \sqrt{3/4\pi}\, y/r$, $Y_1^0(\hat{r}) = \sqrt{3/4\pi}\, z/r$, $Y_1^1(\hat{r}) = \sqrt{3/4\pi}\, x/r$. These transform exactly like the components of a 3D vector under rotation, as required by the $\ell = 1$ irrep.

The Clebsch–Gordan Product

Given equivariant features of angular momenta ℓ_1 and ℓ_2, their tensor product decomposes into a direct sum of irreps:

$$V^{\ell_1} \otimes V^{\ell_2} = \bigoplus_{\ell=|\ell_1-\ell_2|}^{\ell_1+\ell_2} V^\ell \tag{2}$$

The Clebsch–Gordan coefficients $\langle \ell_1 m_1, \ell_2 m_2 | \ell m \rangle$ give the explicit decomposition. This is the $\mathrm{SO}(3)$-equivariant analogue of multiplication: the product of two equivariant features is itself equivariant, with type determined by the triangle rule $|\ell_1 - \ell_2| \le \ell \le \ell_1 + \ell_2$.

The computational cost of tensor products scales as $O(\ell_{\max}^3)$ per interaction—the primary bottleneck in high-ℓ equivariant networks. AlphaFold's design choice to use rigid frames instead of spherical harmonics is motivated by this cost: frames achieve $\mathrm{SE}(3)$-equivariance without tensor products, reducing the computational complexity significantly.

Cross-Reference to Other Parts. The representation theory of $\mathrm{SO}(3)$ used here is the continuous analogue of the representation theory of finite groups used in harmonic analysis and the Fourier transform. The Fourier decomposition on the torus $\mathbb{Z}/p\mathbb{Z}$, which appears

in the grokking analysis of Part VI (Chapter 2), is the discrete version of the same decomposition: characters of the cyclic group play the role of spherical harmonics. Both encode the same principle: a function on a group decomposes into components that transform under irreducible representations.

AlphaFold 2: Architecture and Mathematical Foundations

Historical and Motivational Context. *The AlphaFold2 architecture was designed under two competing constraints: (1) it must be $SE(3)$-equivariant; (2) it must be computationally tractable for proteins of length up to 1000 residues. The tension between these constraints drove the key design decision: rather than using spherical harmonics (which are exact but $O(\ell_{max}^3)$ expensive), AlphaFold2 uses rigid frames to achieve equivariance. The structure module's backbone update represents each residue as a rigid body (rotation + translation) and applies $SE(3)$-equivariant operations in the local coordinate frame of each residue—the Invariant Point Attention (IPA). This design was novel: previous equivariant networks ($SE(3)$-Transformers, TFN) used spherical harmonics and were too slow for protein-scale inputs.*

The Dual Representation: MSA and Pair Features

AlphaFold2 represents a protein with N residues using two tensors:

- The MSA representation $m \in \mathbb{R}^{S \times N \times d_{\text{msa}}}$: S homologous sequences, capturing evolutionary co-variation.
- The pair representation $z \in \mathbb{R}^{N \times N \times d_{\text{pair}}}$: one feature vector per pair of residues, ultimately encoding pairwise distances and orientations.

This dual representation is motivated by the co-evolutionary model: positions that mutate together in evolution tend to be in contact in 3D structure (because they interact physically). The MSA captures which positions co-vary; the pair representation translates this into spatial proximity information.

The Evoformer: Axial Attention and Triangular Updates

The Evoformer block updates the MSA and pair representations through a sequence of operations:

MSA row-wise gated attention with pair bias:

$$a_{ij} = \text{softmax}\left(\frac{Q_i K_j^T}{\sqrt{d}} + b_{ij}\right), \quad b_{ij} = W_b\, z_{ij} \tag{3}$$

Each sequence row attends to other positions in the same sequence, with the attention logit biased by the pair representation. This allows the pair representation to *guide* which MSA positions attend to each other.

MSA column-wise gated attention: Each column (all sequences at one position) attends to other sequences, capturing co-evolutionary signal across the MSA.

Outer product mean: Updates the pair representation from the MSA: $z_{ij} \mathrel{+}= \text{mean}_s(m_{is} \otimes$

m_{js}) (outer product of MSA features at positions i and j, averaged over sequences). This is the key operation that translates MSA co-evolutionary information into pair representation structure.

Theorem 2.1 (Triangular Updates as Geometric Invariants)**.** *The triangular multiplicative update implements a constraint from metric geometry: for residues A, B, C, the distance $d(A, C)$ is approximately determined by $d(A, B)$ and $d(B, C)$ via the triangle inequality. The outgoing triangular update:*

$$\Delta z_{ij} = \sigma(g_{ij}) \odot \sum_k W_a z_{ik} \odot W_b z_{kj} \tag{4}$$

accumulates evidence about the i-j distance from all "bridge" residues k, implementing an approximate triangle inequality constraint. The two variants (incoming/outgoing) cover both $A \to B \to C$ and $A \leftarrow B \to C$ patterns.

Rigid Frames and SE(3)-Equivariance

AlphaFold2 avoids spherical harmonics by using rigid frames. A frame for residue i is an element $T_i = (R_i, t_i) \in \mathrm{SE}(3)$, where:

- $t_i = x_{C\alpha_i}$ (the $C\alpha$ atom position)
- R_i is defined by the N-$C\alpha$-C backbone atoms via Gram–Schmidt orthogonalisation of the $N \to C\alpha$ and $N \to C$ vectors.

An SE(3)-equivariant operation on frames satisfies: $f(g \circ T_1, \dots, g \circ T_n) = g \circ f(T_1, \dots, T_n)$ for any $g \in \mathrm{SE}(3)$, where the composition of frames is $(R_1, t_1) \circ (R_2, t_2) = (R_1 R_2, R_1 t_2 + t_1)$.

Invariant Point Attention (IPA): Full Derivation

IPA computes attention using both sequence-space features and 3D-space coordinates. For residue i with frame $T_i = (R_i, t_i)$:

Local point generation. Each query head h generates a set of query points $\tilde{q}_{ih} \in \mathbb{R}^3$ and key points $\tilde{k}_{jh} \in \mathbb{R}^3$ in the local frame of each residue. In global coordinates, these are: $\hat{q}_{ih} = R_i \tilde{q}_{ih} + t_i, \quad \hat{k}_{jh} = R_j \tilde{k}_{jh} + t_j$.

Invariant attention logit. The attention weight between residues i and j:

$$a_{ij} = \mathrm{softmax}\left(\frac{1}{\sqrt{d_h + \gamma_q}} \left[q_i^T k_j + b_{ij} - \frac{\gamma_q}{2} \sum_h \|\hat{q}_{ih} - \hat{k}_{jh}\|^2 \right] \right) \tag{5}$$

The first term is standard dot-product attention in feature space. The second term is the pair bias from the pair representation. The third term is the sum of squared Euclidean distances between query and key points in global coordinates. This distance term is SE(3)-invariant:

Theorem 2.2 (IPA SE(3) Invariance)**.** *The attention weights a_{ij} are SE(3)-invariant: for any global $g = (R_{global}, t_{global}) \in \mathrm{SE}(3)$, applying g to all frames $T_i \mapsto g \circ T_i$ leaves a_{ij} unchanged.*

Proof. Under g: $\hat{q}_{ih} \mapsto R_{\text{global}}\hat{q}_{ih} + t_{\text{global}}$ and similarly for $\hat{k}_{jh}$. The squared distance: $\|R_{\text{global}}(\hat{q}_{ih} - \hat{k}_{jh})\|^2 = \|\hat{q}_{ih} - \hat{k}_{jh}\|^2$ since rotation preserves norms. The translation cancels: $(R_{\text{global}}\hat{q}_{ih} + t) - (R_{\text{global}}\hat{k}_{jh} + t) = R_{\text{global}}(\hat{q}_{ih} - \hat{k}_{jh})$. □

Backbone update. The IPA output is used to update the residue frames via the backbone frame update operator, which predicts a rigid body transformation ΔT_i to be applied to the current frame, progressively refining the 3D structure over multiple IPA layers.

The FAPE Loss Function

The Frame Aligned Point Error (FAPE) measures structural accuracy independently of the global orientation:

$$\mathcal{L}_{\text{FAPE}} = \frac{1}{NT} \sum_i \sum_{x_p} \|T_i^{-1} x_p - \hat{T}_i^{-1} x_p\|_2 \tag{6}$$

where T_i is the predicted frame, $\hat{T}_i$ is the true frame, and x_p ranges over all atom positions. The frame-relative measurement is more informative than global RMSD: errors at early residues do not propagate into measurements at later residues.

The clipped variant $\min(\|T_i^{-1}x_p - \hat{T}_i^{-1}x_p\|_2, d_{\text{clamp}})$ prevents individual large errors from dominating the gradient—a key practical detail for stable training on PDB structures with occasional erroneous coordinates.

Cross-Reference to Other Parts. The FAPE loss is the structure prediction analogue of the InfoNCE loss in self-supervised learning (Part IV): both are geometric losses that measure proximity in a representation space (Euclidean 3D for FAPE, hypersphere for InfoNCE) in a way that is invariant to nuisance transformations (global SE(3) for FAPE, augmentation for InfoNCE). The backbone frame update is a learned exponential map on the Lie group SE(3)—the continuous analogue of the discrete parameterisation discussed in Part I.

AlphaFold 3: Diffusion for Biomolecular Structure

Historical and Motivational Context. *AlphaFold2's structure module predicts a single structure by iteratively refining backbone torsion angles. This deterministic prediction is appropriate for well-structured proteins but fails for: intrinsically disordered proteins (IDPs, ∼30% of the human proteome), flexible loops, alternative conformations, and non-protein molecules (ligands, nucleic acids, post- translational modifications). AlphaFold3 (Abramson et al., 2024) addressed these limitations by replacing the structure module with a diffusion model that directly generates 3D atom coordinates from a continuous noisy distribution. The key insight: diffusion naturally handles multimodality (multiple valid structures), does not require predefined torsion angle parameterisation, and generalises straightforwardly to arbitrary molecular systems.*

The Pairformer and Single Representation

AlphaFold3 replaces the Evoformer with the Pairformer: a simplified architecture processing the pair representation $z \in \mathbb{R}^{N \times N \times d}$ and single representation $s \in \mathbb{R}^{N \times d}$. The

Pairformer uses:

- Triangle attention on the pair representation (same as Evoformer)
- Triangle multiplicative updates (same as Evoformer)
- Pair transition: a two-layer MLP applied independently to each pair
- Single representation updates via column attention over the pair

For non-protein inputs (small molecules, nucleotides), the MSA is replaced by a single-sequence embedding, making AlphaFold3 applicable to any molecular system with known atomic connectivity.

The Diffusion Module: SE(3)-Equivariant Denoising

Given the pair and single representations from the Pairformer, the diffusion module denoises 3D atom positions. The forward process: $x_t = x_0 + \sigma_t \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, I)$

The diffusion network $D_\theta(x_t, t, s, z)$ takes noisy coordinates, noise level, and Pairformer embeddings, and predicts the clean coordinates $\hat{x}_0$. Training objective:

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{t,\varepsilon}\left[\lambda(t)\, \| D_\theta(x_0 + \sigma_t \varepsilon, t, s, z) - x_0 \|^2\right] \tag{7}$$

The diffusion network must be SE(3)-equivariant: rotating the input coordinates rotates the predicted coordinates correspondingly. This is implemented via an Atom Transformer using IPA-style invariant point attention conditioned on the Pairformer embeddings.

Theorem 3.1 (Diffusion as Conditional Structure Generation)**.** *The diffusion module learns the conditional distribution $p(x_0 | \text{sequence}, \text{MSA}, \text{templates})$. Multiple samples from this distribution correspond to multiple valid 3D structures, capturing conformational diversity that AlphaFold2's single-point prediction cannot represent.*

Cross-Reference to Other Parts. The diffusion module of AlphaFold3 is a direct application of the DDPM framework of Part IV (Chapter 5), with the key difference that the data being denoised is 3D atom coordinates rather than image pixels, and the network must be SE(3)-equivariant rather than translation-equivariant. The probability flow ODE (Part IV, Theorem 4.1) applies here as well: the reverse diffusion can be made deterministic by setting $\sigma_t = 0$ in the DDIM update, enabling fast sampling of single structures while retaining the ability to sample multiple conformations by varying the initial noise.

ESM-3: A Multimodal Protein Language Model

Historical and Motivational Context. *Protein language models—neural networks trained to predict masked or next amino acids in protein sequences—were introduced by Rives et al. (2021) with ESM-1b (650M parameters, trained on UniRef50). The key observation was that these models, despite never seeing 3D structures during training, encode structural information implicitly: the attention patterns correlate with residue-residue contacts in 3D structure. ESM-2 (Lin et al., 2023) scaled to 15B parameters and demonstrated that sequence-only models approach AlphaFold2*

accuracy when combined with the Evoformer. ESM-3 (Hayes et al., 2024) extended the model to jointly model sequence, structure, and function, enabling the generation of new proteins that satisfy specified structural and functional constraints.

Tokenisation of Three Modalities

ESM-3 treats protein sequence, structure, and function as three separate "tracks" of discrete tokens:

Sequence tokens: 20 amino acid tokens plus special tokens. Standard discrete vocabulary, same as BERT/GPT-style language models.

Structure tokens: 3D coordinates are tokenised via a VQ-VAE trained to quantise structural neighbourhoods. Each residue's local geometric environment (its 3D neighbourhood of ± 16 residues) is encoded as a single discrete token from a vocabulary of $\sim$4096 structure tokens. The VQ-VAE uses the same quantisation mechanism as described in Part IV (Chapter 8), with the straight-through gradient for training.

Function tokens: InterPro annotations and Gene Ontology terms are encoded as binary bit vectors (function tokens), one per known functional family.

The three token types are embedded into a shared latent space. A geometric attention module processes the structure tokens in a geometry-aware manner, analogous to IPA but adapted for discrete structure token representations.

The Unified Masked Language Model

ESM-3 is trained with masked language modelling simultaneously across all three modalities. The training objective:

$$\mathcal{L}_{\text{ESM-3}} = \mathbb{E}\left[-\sum_{\text{masked } i} \log p_\theta(x_i^S, x_i^T, x_i^F | \text{context})\right] \tag{8}$$

where (x_i^S, x_i^T, x_i^F) are the sequence, structure, and function tokens at position i and context denotes all unmasked tokens.

Theorem 4.1 (Generative Capacity of Multimodal MLM)**.** *ESM-3 trained as a joint masked LM learns the joint distribution $p(S, T, F)$ and all conditionals $p(S|T, F)$, $p(T|S, F)$, $p(F|S, T)$, etc. simultaneously. Any conditional can be evaluated by setting appropriate modalities as context. This enables: structure prediction (sample $p(T|S)$), inverse folding (sample $p(S|T)$), function prediction (sample $p(F|S)$), and unconditional generation (sample $p(S, T, F)$)—all from a single trained model.*

Cross-Reference to Other Parts. ESM-3's multimodal masked LM is the protein analogue of the unified multimodal architectures (Gemini, GPT-4V) discussed in Part VI (Chapter 6): both treat multiple modalities as token sequences in a single autoregressive or masked model. The VQ-VAE structure tokenisation directly applies the framework of Part IV (Chapter 8). The cross-modal conditioning—predicting sequence from structure—is the protein analogue of CLIP's cross-modal alignment: both learn a shared embedding space where semantically related objects (sequence and its fold; image and its caption) are

aligned.

AlphaGenome: Deep Learning for the Genome

Historical and Motivational Context. *Predicting gene expression and regulatory activity from DNA sequence has been a central goal of computational biology since the genome era. Early methods used position weight matrices to detect transcription factor binding sites; deep learning approaches (DeepBind, 2015; DeepSEA, 2015) showed that CNNs could predict epigenetic signals directly from sequence. Enformer (Avsec et al., 2021) extended the sequence-to-function approach to 200kb context windows using a transformer, achieving state-of-the-art predictions across hundreds of gene expression and chromatin accessibility tracks. AlphaGenome extends this to million-base-pair context windows with multi-scale architecture.*

Multi-Scale Genomic Tokenisation

Raw DNA sequences of length $L = 10^5$–10^6 nucleotides are processed through a multi-scale convolution stem:

$$\begin{aligned} h^{(1)} &= \text{Conv}_{k=15}(x_{\text{nuc}}), \quad \text{dim: } L \to L \\ h^{(k)} &= \text{Pool} + \text{Conv}(h^{(k-1)}), \quad \text{dim: } L/4^{k-1} \to L/4^k \end{aligned} \tag{9}$$

Cross-scale attention allows each resolution level to attend to other scales, enabling long-range genomic interactions (enhancer-promoter contacts at distances of 100kb) to influence local regulatory predictions.

Watson-Crick Complementarity as a Symmetry Constraint

Double-stranded DNA has a fundamental symmetry: the reverse complement operation $\text{RC} : x_1 x_2 \cdots x_L \mapsto c(x_L)c(x_{L-1}) \cdots c(x_1)$, where $c(A) = T$, $c(T) = A$, $c(C) = G$, $c(G) = C$.

AlphaGenome enforces reverse complement equivariance: $f(\text{RC}(x)) = \text{RC}(f(x))$. The practical implementation trains on both x and $\text{RC}(x)$, averaging predictions—equivalent to constructing an equivariant model by symmetrisation. The weight sharing induced by this symmetry reduces the effective parameter count by $\sim 2\times$ and improves sample efficiency.

Multi-Task Prediction and Gradient Conflict Resolution

AlphaGenome simultaneously predicts k regulatory phenotypes (RNA-seq, ATAC-seq, ChIP-seq, Hi-C, splicing) from shared representations. The multi-task loss: $\mathcal{L} = \sum_t \lambda_t \mathcal{L}_t$, $\lambda_t \propto 1/\|\nabla_\theta \mathcal{L}_t\|$ (approximate GradNorm, equalising gradient magnitudes across tasks).

PCGrad (Yu et al., 2020) resolves gradient conflict between tasks whose optimal directions point against each other:

$$\hat{g}_t = g_t - \frac{g_t \cdot g_{t'}}{\|g_{t'}\|^2} g_{t'} \quad \text{if} \quad g_t \cdot g_{t'} < 0 \tag{10}$$

eliminating the component of g_t that conflicts with $g_{t'}$.

Equivariant GNNs for Molecular Property Prediction

Historical and Motivational Context. *Graph neural networks for molecules were introduced before equivariance became the central design principle. Duvenaud et al. (2015) and Gilmer et al.'s message passing neural network (MPNN, 2017) treated molecules as graphs with atom features, but used only connectivity information (bond distances as scalars), not full 3D geometry. SchNet (Schütt et al., 2017) was the first to incorporate continuous pairwise distances into a differentiable GNN. The progression toward higher geometric expressivity—DimeNet (angles), PaiNN (equivariant vectors), NequIP/MACE (equivariant tensors)—tracked the theoretical understanding of what information is needed for chemical accuracy.*

The Expressivity Hierarchy: SchNet to MACE

The progression from SchNet to MACE illustrates how successive incorporation of geometric information improves expressivity:

SchNet (Schütt et al., 2017): uses only pairwise distances $d_{ij} = \|r_i - r_j\|$. Message: $m_{ij} = \phi(h_i, h_j, d_{ij})$. Cannot distinguish mirror images (chirality).

DimeNet (Klicpera et al., 2020): uses distances and angles θ_{jik}. Message incorporates the triplet $(d_{ij}, d_{jk}, \theta_{jik})$. Distinguishes some non-isomorphic graphs that SchNet cannot.

PaiNN (Schütt et al., 2021): equivariant vector features alongside scalars. Each atom has scalar state $s_i \in \mathbb{R}^d$ and vector state $v_i \in \mathbb{R}^{3\times d}$. Equivariant message: $\Delta v_i = \sum_j \phi_v(s_i, s_j, d_{ij}) \cdot (r_j - r_i)/d_{ij}$

NequIP/MACE: full equivariant tensor features at each atom, updated via Clebsch–Gordan tensor products.

Theorem 6.1 (Expressivity Hierarchy of Geometric GNNs)**.** *The following strict expressivity hierarchy holds for molecular graphs with 3D coordinates:*

$$\textit{SchNet} \subsetneq \textit{DimeNet} \subsetneq \textit{PaiNN} \subsetneq \textit{NequIP/MACE} \tag{11}$$

Specifically: (i) there exist pairs of molecules indistinguishable by SchNet but distinguishable by DimeNet; (ii) there exist chiral pairs indistinguishable by DimeNet but distinguishable by PaiNN; (iii) accurate force prediction requires tensor-equivariant architectures. The cost of higher expressivity is $O(\ell_{max}^3)$ per tensor product, motivating MACE's body-order expansion as an efficient approximation.

E(n)-Equivariant GNNs

EGNN (Satorras et al., 2021) achieves SE(3)-equivariance without spherical harmonics by directly updating coordinate positions:

$$\begin{aligned} m_{ij} &= \phi_e(h_i, h_j, \|x_i - x_j\|^2) \\ x_i' &= x_i + \sum_j (x_i - x_j)\,\phi_x(m_{ij}) \\ h_i' &= \phi_h(h_i, \sum_j m_{ij}) \end{aligned} \tag{12}$$

The coordinate update $\sum_j (x_i - x_j)\,\phi_x(m_{ij})$ is equivariant: translating all coordinates by t translates x_i' by t; rotating all coordinates by R rotates x_i' by R (since the difference $x_i - x_j$ transforms equivariantly). EGNN achieves equivariance without tensor products, making it more computationally efficient than NequIP/MACE at the cost of limited expressivity (no higher-ℓ features).

Cross-Reference to Other Parts. The equivariant GNNs discussed here are the molecular science analogue of the convolutional neural networks analysed in Part II: both achieve equivariance by weight sharing along the relevant symmetry group (S_n for permutations, $\mathbb{Z}^2$ for translations in CNNs, SE(3) for molecular rotation). The NequIP/MACE tensor product operations are the molecular analogue of the self-attention mechanism in transformers (Part II): both compute pairwise interactions between all atoms (or tokens) and aggregate them into updated representations, but the molecular case requires the interactions to be equivariant rather than merely permutation-equivariant.

Equivariant Message Passing: NequIP, MACE, and the Body-Order Expansion

Historical and Motivational Context. *The combination of spherical harmonics with graph neural networks—equivariant message passing—was introduced by Thomas et al. (2018) in the Tensor Field Networks (TFN), and independently by Fuchs et al. (2020) in the SE(3)-Transformer. These models were theoretically sound but computationally expensive: tensor products scale as $O(\ell_{max}^6)$ per edge when the full CG decomposition is used. NequIP (Batzner et al., 2022) showed that restricting to $\ell \leq 3$ and using optimised CG code achieves chemical accuracy on force fields—matching the accuracy of DFT calculations at a fraction of the cost. MACE (Batatia et al., 2022) further improved efficiency by factoring the message construction into a body-order expansion, enabling higher-order interactions at reduced cost.*

NequIP: Neural Equivariant Interatomic Potentials

NequIP (Batzner et al., 2022) builds an SE(3)-equivariant GNN by:

1. Initialising each atom i with a scalar feature $h_i^{(0)} = \text{embed}(Z_i)$ (embedding of atomic number).
2. Computing edge embeddings $e_{ij} = \text{RadialBasis}(\|r_i - r_j\|) \otimes Y^\ell(\hat{r}_{ij})$: a radial basis function of the distance times the spherical harmonic of the direction.

3. Updating atom features via equivariant message passing: $m_i^{(k)} = \sum_{j \in \mathcal{N}(i)} W_{\text{msg}}^{(k)} \otimes_{CG} h_j^{(k)} \otimes_{CG} e_{ij}$
4. After L message-passing layers, the energy is: $E = \sum_i \text{MLP}(h_i^{(L),\ell=0})$ (only the invariant $\ell = 0$ component), and forces are obtained by automatic differentiation: $F_i = -\partial E / \partial r_i$.

NequIP achieves mean force errors of $\sim$10 meV/Å on MD-17 benchmarks, matching CCSD(T) reference calculations at $\sim$1000$\times$ lower cost.

MACE: Many-Body Atomic Cluster Expansion

MACE (Batatia et al., 2022) introduces the body-order expansion to control the order of many-body interactions efficiently. The message at order ν involves ν neighbours simultaneously:

$$A_i^{(\nu)} = \sum_{j_1, \ldots, j_\nu \in \mathcal{N}(i)} \phi(h_{j_1}, r_{ij_1}) \otimes_{CG} \cdots \otimes_{CG} \phi(h_{j_\nu}, r_{ij_\nu}) \tag{13}$$

Direct computation costs $O(|\mathcal{N}|^\nu)$, but MACE exploits the *symmetric contraction* to reduce this: $A_i^{(\nu)} = \langle \Phi_i^\nu \rangle$ where $\Phi_i = \sum_j \phi(h_j, r_{ij})$ is a single-neighbour sum and the average over ν copies corresponds to computing the symmetrised tensor power $(\Phi_i)^{\otimes \nu}$. This reduces cost to $O(|\mathcal{N}|)$ per order, with the body-order controlled by $\nu_{\max}$.

Theorem 7.1 (MACE Universality). *MACE with body order $\nu \geq 2$ and $\ell_{max} \geq 1$ is a universal approximator for $SE(3)$-invariant functions on atomic environments: for any continuous $SE(3)$-invariant function f and $\varepsilon > 0$, there exist MACE parameters achieving $\|f(\cdot) - f_{MACE}(\cdot)\|_\infty \leq \varepsilon$. The universality is achieved through the combination of radial basis functions (distance universality), spherical harmonics (angular universality), and the body-order expansion (interaction order universality).*

Protein Language Models: Scaling and Evolutionary Information

Historical and Motivational Context. *Protein language models were inspired by the success of BERT and GPT on text, but the analogy between amino acid sequences and sentences is deeper than metaphor: both are generated by an evolutionary or generative process that encodes structured information in sequential form. The key difference is that amino acid sequences are strongly constrained by the fitness of the encoded protein—random sequences almost never fold into stable structures. This means that the distribution of natural protein sequences is a highly compressed representation of the space of possible protein structures, and a model that learns this distribution implicitly encodes structural information.*

ESM-2: Scaling Laws for Protein Sequences

ESM-2 (Lin et al., 2023) is a protein language model trained on UniRef50 (250 million sequences) with masked language modelling, scaled from 8M to 15B parameters. Key findings:

- **Structural information scales:** contact map prediction accuracy (measuring how well attention maps predict which residues are in 3D contact) improves monotonically with model size, reaching near-AlphaFold2 accuracy at 650M parameters for well-studied protein families.
- **Scaling law:** perplexity on held-out sequences follows a power law in model parameters similar to the Chinchilla law for language models.
- **Representations transfer:** ESM-2 embeddings provide competitive starting points for structure prediction (ESMFold) and protein design tasks.

ESMFold: Structure Prediction from Sequence Embeddings

ESMFold (Lin et al., 2023) feeds ESM-2 embeddings directly into a structure prediction network without MSA, achieving AlphaFold2-comparable accuracy on easy targets with 60× faster inference. The architecture:

1. ESM-2 (650M) embeds the sequence into per-residue representations.
2. A folding trunk (similar to the AlphaFold2 structure module) takes these representations and iteratively refines backbone frames.
3. The frames are converted to 3D coordinates via the backbone update operator.

ESMFold's advantage is inference speed: AlphaFold2 requires MSA computation (minutes for large proteins), while ESMFold requires only a single forward pass (seconds). This enables proteome-scale structure prediction for all ∼200M known proteins.

Cross-Reference to Other Parts. ESMFold's use of a pre-trained language model (ESM-2) as a feature extractor for the structure prediction network is an instance of the transfer learning paradigm studied in Part VI (Chapter 5): the language model is fine-tuned (the ESM-2 weights are frozen in ESMFold, which is equivalent to LoRA with rank 0). The scaling laws for ESM-2 follow the same Chinchilla-like power laws documented in Part VI (Chapter 4), confirming that the scaling behaviour of neural networks is a universal property independent of the data modality.

Drug Discovery and Molecular Generation

Historical and Motivational Context. *Drug discovery is a canonical application of generative AI: the goal is to design new molecules with specified properties (binding affinity to a target, ADMET properties, synthetic accessibility). The chemical space of drug-like molecules is estimated at 10^{23}–10^{60} compounds, far too large for exhaustive search. ML-based generative models (Gómez-Bombarelli et al., 2018) began with VAE-based methods in SMILES space; the field progressed through graph-based generation (JT-VAE), flow-based generation (GraphNVP), and equivariant 3D generation (GEOM-Drugs, EDM). The key challenge is the need for 3D generation: drug binding depends critically on the 3D shape of the molecule, not just its graph structure.*

Equivariant Diffusion Models for Molecules

EDM (Equivariant Diffusion Models, Hoogeboom et al., 2022) extends DDPM to generate 3D molecular coordinates and atom types simultaneously:

- **Forward process:** adds Gaussian noise to both coordinates $x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\varepsilon_x$ and categorical atom types h_t (via absorbing diffusion or multinomial noise).
- **Reverse process:** an equivariant GNN denoises both coordinates and atom types simultaneously.
- **Equivariance:** the denoising network $\varepsilon_\theta(x_t, h_t, t)$ is SE(3)-equivariant, ensuring that the generated molecule is physically valid regardless of the initial noise orientation.

The key theorem from EDM: the DDPM ELBO applied to the joint distribution (x, h) factorises as: $\mathcal{L}_{\text{ELBO}} = \mathcal{L}_{\text{coord}} + \mathcal{L}_{\text{atom}}$ if the coordinate and atom type noising processes are independent, enabling separate training of the two denoising objectives.

Property-Guided Generation and Classifier Guidance

Generating molecules with specified properties uses classifier guidance (analogous to classifier-free guidance for images, Part IV): $\nabla_{x_t} \log p(x_t|y) = \nabla_{x_t} \log p_t(x_t) + \gamma \nabla_{x_t} \log p(y|x_t)$ where y is the desired property (binding affinity, solubility). The property predictor $p(y|x_t)$ must operate on noisy coordinates; in practice, a separate property network trained on noisy molecules is used.

> **Cross-Reference to Other Parts.** The equivariant diffusion model for molecules is the intersection of two frameworks from this series: the DDPM diffusion process of Part IV (Chapter 5) and the SE(3)-equivariant architecture of Chapter 1 in this part. The classifier guidance for property-directed generation is the molecule generation analogue of the RLHF guidance for language generation (Part V, Chapter 5): in both cases, a reward/property signal steers the generative process toward desired outputs, trading diversity for constraint satisfaction.

Quantum Chemistry and Foundation Models for Materials

From DFT to Machine Learning Potentials

Density Functional Theory (DFT) solves the Schrödinger equation approximately for electronic structure, providing quantum-mechanical ground truth for molecular energies and forces. DFT scales as $O(N^3)$ in the number of electrons, limiting it to $\sim$1000 atoms. Machine learning potentials (MLPs) learn to approximate the DFT potential energy surface from a training set of DFT calculations, achieving $O(N)$ scaling.

The fundamental theorem justifying MLPs is the Hohenberg-Kohn theorem (1964): the ground-state energy of an N-electron system is a unique functional of the electron density $\rho(\mathbf{r})$. Since $\rho(\mathbf{r})$ is itself determined by the nuclear positions $\{R_i\}$, the total energy $E(\{R_i\})$ is a function of nuclear positions alone—the potential energy surface that MLPs approximate.

Materials Project and Universal Force Fields

The Materials Project (Jain et al., 2013) has computed DFT properties of >150,000 inorganic crystals, creating the largest open database of quantum-mechanical calculations. Universal Neural Network Force Fields trained on the Materials Project (M3GNet, CHGNet, MACE-MP-0) achieve near-DFT accuracy across the periodic table—the first "universal" force fields that work for any inorganic material without refitting.

Cross-Reference to Other Parts. Universal force fields trained on the Materials Project represent the "foundation model" paradigm (Part VI) applied to materials science: pretraining on large diverse datasets followed by fine-tuning for specific applications. The transfer learning framework of Part VI (Chapter 5)—LoRA, QLoRA—applies directly: adapting MACE-MP-0 to a specific material family requires only fine-tuning the last few layers on a small dataset of DFT calculations for that family.

Geometric Deep Learning: The Unifying Blueprint

Historical and Motivational Context. *The Geometric Deep Learning (GDL) framework (Bronstein, Bruna, LeCun, Szlam, Vandergheynst, 2017; Bronstein et al., 2021) provided a unified mathematical language for understanding CNNs, RNNs, GNNs, and equivariant networks as special cases of the same blueprint: choose a symmetry group G acting on a domain Ω, choose G-equivariant signal processing operations, and stack them. This unification, while primarily pedagogical, had practical consequences: it showed that every good deep learning architecture encodes a specific prior about the symmetries of the problem, and that designing new architectures reduces to identifying the correct symmetry group.*

The GDL Blueprint: Domain, Symmetry, and Signal

The GDL framework specifies four components:

1. **Domain** Ω: the structured space the data lives on. Examples: grid $\mathbb{Z}^2$ (images), sequence $\{1, \dots, L\}$ (text), graph $G = (V, E)$ (molecules), manifold $\mathcal{M} \subset \mathbb{R}^3$ (surfaces).
2. **Symmetry group** $\mathfrak{G}$: the group of transformations of Ω that are symmetries of the task. Examples: translations $\mathbb{Z}^2$ (images), permutations S_n (sets/text), graph automorphisms (graph classification), SE(3) (molecules in 3D).
3. **Signal** $\mathcal{X}(\Omega)$: functions on the domain. A $\mathfrak{G}$-signal is a function $f : \Omega \to \mathbb{R}^c$ equipped with a group action $(\rho(g)f)(u) = f(g^{-1}u)$.
4. **Equivariant layers**: mappings between signals that commute with the group action.

Theorem 11.1 (Universal Approximation for Equivariant Functions)**.** *For a compact group G acting on $\mathbb{R}^d$, any continuous G-equivariant function $f : \mathbb{R}^d \to \mathbb{R}^d$ can be approximated arbitrarily well by a composition of: (1) G-equivariant linear layers (parameterised by Schur's Lemma), (2) elementwise nonlinearities, and (3) G-invariant pooling operations. The approximation error is bounded by the network depth and width via the same universal approximation*

argument as for standard networks.

Special Cases of the GDL Blueprint

Each major deep learning architecture corresponds to a specific instance:

Convolutional Neural Networks: $\Omega = \mathbb{Z}^2$ (pixel grid), $\mathfrak{G} = \mathbb{Z}^2$ (discrete translations), equivariant layers = convolutions $[W * f](x) = \sum_u W(u - x) f(u)$.

Transformers: $\Omega = \{1, \dots, L\}$ (sequence positions), $\mathfrak{G} = S_L$ (permutations), equivariant maps = set functions (attention is S_L-equivariant; the transformer adds positional encodings to break the permutation symmetry deliberately).

Graph Neural Networks: $\Omega = G = (V, E)$, $\mathfrak{G} = \mathrm{Aut}(G)$ (graph automorphisms), equivariant layers = message passing.

SE(3)-equivariant networks (this part): $\Omega = \mathbb{R}^3$, $\mathfrak{G} = \mathrm{SE}(3)$, equivariant layers = tensor field operations or IPA-style invariant attention.

Diffusion for Protein Design and Inverse Problems

RFDiffusion: Protein Backbone Design

RFDiffusion (Watson et al., 2023) applies diffusion directly in the space of protein backbone frames, enabling de novo protein design: generating backbone geometries that fold into stable structures and can be functionalised.

The forward process adds noise to the rotation matrices and translation vectors of each residue frame:

$$\begin{aligned} R_t^i &= R_0^i \,\mathrm{Exp}(\sqrt{t}\,\xi_t^i), \quad \xi_t^i \sim \mathcal{N}(0, I_3) \\ x_t^i &= x_0^i + \sqrt{t}\,\varepsilon_t^i, \quad \varepsilon_t^i \sim \mathcal{N}(0, I_3) \end{aligned} \tag{14}$$

The noise on $\mathrm{SO}(3)$ uses the geodesic distance: rotations are perturbed by random elements of the Lie algebra $\mathfrak{so}(3)$.

The reverse process denoises using a fine-tuned RoseTTAFold network as the denoising backbone, predicting the clean frames R_0^i, x_0^i from the noisy frames R_t^i, x_t^i. The SE(3)-equivariance of the RoseTTAFold network ensures that the generated backbone is physically valid.

Theorem 12.1 (Diffusion on Lie Groups)**.** *The forward-reverse diffusion framework extends to Lie groups G by replacing Gaussian noise on $\mathbb{R}^n$ with the isotropic Gaussian on G: $q(g_t|g_0) = \int_G q(g_t|g_1)q(g_1|g_0)\,dg_1$ with $q(g_t|g_{t-1}) \propto \exp(-d_G(g_t, g_{t-1})^2/(2\sigma_t^2))$, where d_G is the Riemannian distance on G. The score function is $\nabla_{g_t} \log q(g_t) = -\mathrm{Log}_{g_t}(g_t)/\sigma_t^2$, where Log_{g_t} is the Riemannian logarithm map.*

FrameDiff and Protein Motif Scaffolding

FrameDiff (Yim et al., 2023) introduces a pure diffusion framework for protein backbone generation (without the RoseTTAFold prior) using the SE(3) diffusion process.

The model learns to generate backbone geometries consistent with stable protein folding via a score-based reverse process on the product space $(\mathrm{SO}(3))^N \times (\mathbb{R}^3)^N$ for an N-residue protein.

Motif scaffolding conditions on a subset of fixed residues (the motif) and generates the surrounding scaffold: $p(\text{scaffold}|\text{motif}, N)$. This enables functional site preservation: generating new proteins that carry an active site with specified 3D geometry.

Cross-Reference to Other Parts. The diffusion on SE(3) used in RFDiffusion/FrameDiff is the geometric generalisation of the DDPM/DDIM framework of Part IV (Chapter 5). The Riemannian score function—the gradient of the log-density on a manifold—connects to the information geometry of Part I: the score function on the statistical manifold is the gradient in the Fisher metric, and the score function on SO(3) is the gradient in the bi-invariant metric on the Lie group. The connection between the two is provided by the exponential family structure: probability distributions on SO(3) form an exponential family whose natural statistics are the characters of the irreducible representations.

Multi-Scale Genomic Architecture: Mathematical Details

Hyena Operators for Long-Range Genomic Context

The quadratic attention cost $O(L^2)$ limits transformer context to ∼100k nucleotides, but regulatory interactions can span megabases. Hyena (Poli et al., 2023) replaces attention with a recurrent long convolution parameterised by a state-space model:

$$y_n = (h * x)_n = \sum_{k=0}^{L-1} h_k \, x_{n-k} \tag{15}$$

where h_k is an implicit convolution kernel (parameterised via a small neural network, not stored explicitly). Hyena computes this in $O(L \log L)$ via FFT, enabling genomic sequence modelling at megabase scale.

The Mamba architecture (Gu and Dao, 2023) extends this with selective state spaces: the state-space matrices A, B, C, D depend on the input x, enabling the model to "select" what information to retain in the state:

$$\begin{aligned} h_t &= A h_{t-1} + B x_t \\ y_t &= C h_t + D x_t \end{aligned} \tag{16}$$

where A, B, C, D are input-dependent. Mamba achieves linear-time inference (no quadratic attention) with quality competitive to transformers on language, DNA, and audio tasks.

Caduceus: Bidirectional Mamba for Genomics

DNA is bidirectional: regulatory elements can act on downstream genes (from 5′ to 3′, forward strand direction) or upstream genes (from the reverse complement strand). Caduceus (Schiff et al., 2024) extends Mamba to bidirectional sequence modelling for DNA by running two independent Mamba models in opposite directions and combining their states: $y_i = \mathrm{MLP}(h_i^{\text{fwd}}, h_i^{\text{rev}})$.

The reverse complement equivariance (Chapter 5) is enforced architecturally: the reverse direction model receives the reverse complement of the input sequence, ensuring that the model's predictions for the forward strand sequence are identical to those for its reverse complement.

Symmetry Breaking and Chiral Molecules

Chirality: When SE(3)-Invariance Is Wrong

Most equivariant molecular models are invariant under full O(3) (including reflections), not just SO(3). But chirality—the handedness of a molecule— is a physical property that changes under reflection. L-alanine and D-alanine are mirror images with different biological activity; one is a natural amino acid, the other is not. An O(3)-invariant model predicts identical properties for both, which is physically incorrect.

SE(3)-equivariant (but not O(3)-equivariant) models can distinguish enantiomers because $\mathrm{SE}(3) = \mathrm{SO}(3) \ltimes \mathbb{R}^3$ does not include reflections. When $\ell = 1$ pseudo-vector features (which flip sign under reflection) are used in the equivariant network, the network output changes under reflection—correctly encoding chirality.

The mathematical distinction: scalars ($\ell = 0$) and symmetric tensors are invariant under O(3); pseudo-vectors ($\ell = 1$ odd-parity) and pseudo-tensors are invariant under SO(3) but change sign under improper rotations (reflections). An SE(3)-equivariant network that uses $\ell = 1$ features can distinguish enantiomers; an O(3)-invariant network that uses only $\ell = 0$ features cannot.

Cross-Reference to Other Parts. The distinction between scalars and pseudo-vectors is the mathematical foundation of the parity violation in physics: the weak nuclear force distinguishes left-handed from right-handed fermions, violating O(3) invariance but preserving SO(3) invariance. The group-theoretic treatment of this distinction in Part VII connects to the representation theory of Part II: the $\ell = 1$ pseudo-vector representation of O(3) is an instance of a projective representation (representation up to a sign), analogous to the spinor representations used in quantum mechanics.

Cryo-EM and Structure Determination from Noisy Projections

Historical and Motivational Context. *Cryo-electron microscopy (cryo-EM) won the 2017 Nobel Prize in Chemistry for enabling atomic-resolution structure determination of proteins too large or flexible for X-ray crystallography. A cryo-EM experiment captures tens of thousands of 2D projections of randomly oriented protein molecules. The 3D structure is reconstructed from these projections—an ill-posed inverse problem (recovering a 3D volume from 2D projections with unknown orientations) that has been transformed by deep learning methods. CryoDRGN (Zhong et al., 2020) introduced variational autoencoder-based reconstruction that simultaneously recovers the 3D structure and its conformational heterogeneity—multiple 3D states of the protein rather than a single rigid structure.*

The Cryo-EM Inverse Problem

A cryo-EM image I is a noisy 2D projection of the 3D electron density map $V : \mathbb{R}^3 \to \mathbb{R}$:

$$I_\phi(\mathbf{r}) = \int V(R_\phi \mathbf{x} + t)\, dx_3 + \eta \tag{17}$$

where $R_\phi \in \mathrm{SO}(3)$ is the (unknown) rotation, $t \in \mathbb{R}^2$ is the (unknown) translation, x_3 is the depth axis, and η is detector noise. Reconstruction requires estimating both V and the poses (R_ϕ, t) for each image—a non-convex optimisation problem in a joint space with millions of images and thousands of volume parameters.

The Fourier slice theorem. The 2D Fourier transform of the projection equals the central slice of the 3D Fourier transform: $\hat{I}_\phi(\mathbf{k}) = \hat{V}(R_\phi[\mathbf{k}; 0])$. This geometric relationship—the projection-slice theorem—is the mathematical foundation of all tomographic reconstruction methods. Each 2D image provides a 2D slice through the 3D Fourier volume, and combining many slices from known orientations recovers the full 3D Fourier volume.

CryoDRGN: Variational Reconstruction with Conformational Heterogeneity

CryoDRGN (Zhong et al., 2020) frames cryo-EM reconstruction as a VAE problem:

- **Encoder:** maps each image I to a latent code $z \in \mathbb{R}^k$ capturing its conformational state (which 3D structure it depicts).
- **Decoder:** an implicit neural representation (INR) that maps any 3D point $\mathbf{x}$ and latent code z to electron density: $V(\mathbf{x}; z) = f_\theta(\mathbf{x}, z)$ parameterised by an MLP.
- **ELBO:** trains both encoder and decoder to maximise: $\mathcal{L} = \mathbb{E}_{q_\phi(z|I)}[\log p_\theta(I|V(\cdot; z), \phi)] - D_{KL}(q_\phi(z|I) \| p(z))$

The decoder's use of an INR enables continuous interpolation in conformational space: traversing the latent space z produces a continuous family of 3D structures, capturing the conformational landscape of the protein.

> **Cross-Reference to Other Parts.** CryoDRGN is the scientific application of the VAE framework of Part IV (Chapter 1): the latent code z plays the same role as the image latent, and the decoder is an SE(3)-aware neural field rather than a convolutional decoder. The implicit neural representation (INR) used for the 3D volume connects to NeRF (Neural Radiance Fields) for 3D scene reconstruction—both use MLPs that map 3D coordinates to a scalar field (density/radiance).

SE(3) Score Matching for Molecular Docking

DiffDock: Diffusion for Protein-Ligand Docking

Molecular docking—predicting the 3D pose of a small molecule in a protein's binding site—is a central problem in drug discovery. DiffDock (Corso et al., 2023) formulates docking as a generalised diffusion problem over the product space of ligand transla-

tions, rotations, and torsion angles:

$$\text{pose} = (t, R, \tau_1, \ldots, \tau_K) \in \mathbb{R}^3 \times \mathrm{SO}(3) \times [0, 2\pi]^K \tag{18}$$

where t is the translation, R is the global rotation, and τ_k are the K rotatable bond torsion angles. The forward diffusion adds independent noise to each component:

- Translation: $t_s = t_0 + \sigma_s \varepsilon_t$, $\varepsilon_t \sim \mathcal{N}(0, I_3)$.
- Rotation: $R_s = \mathrm{Exp}_{R_0}(\sigma_s^R \xi_s)$, $\xi_s \sim \mathcal{N}(0, I_3)$ on $\mathfrak{so}(3)$.
- Torsions: $\tau_s = \tau_0 + \sigma_s^\tau \varepsilon_\tau$ on the torus.

The reverse process denoises all components simultaneously using an $\mathrm{SE}(3)$-equivariant GNN that takes the protein surface and noisy ligand pose as input, predicting the score (denoising direction) for each component.

Confidence Calibration for Docking Predictions

DiffDock generates multiple candidate poses by running the reverse diffusion from different initial noises. A confidence model $C_\phi(\text{pose})$ ranks these candidates, predicting the likelihood that each pose is within 2Å RMSD of the true binding pose. The confidence model is trained on DiffDock's own predictions, creating a self-supervised calibration loop.

This approach achieved a top-1 success rate of 38% on PoseBusters benchmarks versus 23% for GNINA (the prior state-of-the-art AutoDock-based method), representing a significant improvement in the accuracy of computational drug design.

Graph Transformers and Long-Range Interactions

Historical and Motivational Context. *Standard message-passing GNNs are limited in their ability to propagate information across long graph distances: k layers of message passing provide information from at most k-hop neighbourhoods. For molecular property prediction, long-range interactions (e.g. electrostatic effects across the entire molecule) require many layers, but deep GNNs suffer from over-smoothing (all node representations converge to the same vector) and over-squashing (information from exponentially many paths is squeezed through a bottleneck). Graph Transformers address these limitations by allowing each node to attend to all other nodes simultaneously—providing global receptive fields at every layer.*

Graphormer: Adding Graph Structure to Transformers

Graphormer (Ying et al., 2021) augments the standard transformer with graph-structural biases:

$$a_{ij} = \frac{(h_i W_Q)(h_j W_K)^T}{\sqrt{d}} + b_{\phi(i,j)} + c_{ij} \tag{19}$$

where:

- $b_{\phi(i,j)}$ is the spatial encoding: a learnable scalar per graph distance $\phi(i,j) =$ shortest path length between i and j.

- c_{ij} is the edge encoding: a projection of edge features along the shortest path between i and j.

The spatial encoding injects the graph topology into the attention weights, enabling the transformer to leverage graph structure while maintaining the global receptive field of attention.

Equivariant Graph Transformers

Equiformer (Liao and Smidt, 2022) combines transformer global attention with SE(3)-equivariant tensor features. The key operation is equivariant attention:

$$\text{out}_i = \sum_j \underbrace{a(h_i, h_j, r_{ij})}_{\text{SE(3)-inv. weights}} \cdot \underbrace{T_{CG}(h_j, Y(r_{ij}/\|r_{ij}\|))}_{\text{SE(3)-equivariant values}} \tag{20}$$

The attention weights $a(\cdot)$ are SE(3)-invariant (depend only on distances and inner products); the values $T_{CG}(\cdot)$ are SE(3)-equivariant (involve Clebsch-Gordan products with spherical harmonics). This combination achieves the expressivity of tensor-equivariant networks with the global receptive field of transformers.

Cross-Reference to Other Parts. The Graph Transformer framework is the molecular graph analogue of the ViT (Vision Transformer) of Part II: both replace the local receptive field of CNNs/GNNs with the global receptive field of transformer self-attention. The equivariant graph transformer (Equiformer) is the molecular analogue of the IPA mechanism in AlphaFold2 (Chapter 2): both combine invariant scalar attention weights with equivariant value aggregation. The NTK analysis of Part VI applies to graph transformers as well: the NTK of a graph transformer depends on the Gram matrix of node features, and the spectral properties of the graph Laplacian determine the effective depth of the model.

Single-Cell Genomics and Trajectory Inference

Historical and Motivational Context. *Single-cell RNA sequencing (scRNA-seq, Taniguchi et al., 2011; 10X Genomics, 2016) enables measuring gene expression in individual cells, revealing the cellular heterogeneity within tissues that bulk RNA-seq averages away. A single scRNA-seq experiment can profile 10^4–10^5 cells, each described by ∼20,000 gene expression levels—a high-dimensional, sparse data matrix. The key challenge is that cells are destroyed in measurement, so time is not directly observable; instead, cell state trajectories (e.g. stem cell differentiation) must be inferred from the snapshot of cellular states.*

Variational Autoencoders for Single-Cell Data

scVI (Lopez et al., 2018) applies a VAE to model the count distribution of gene expression data. The generative model:

$$\begin{aligned} z &\sim \mathcal{N}(0, I) \\ l &\sim \text{LogNormal}(\ell_\mu, \ell_\sigma^2) \quad \text{(library size)} \\ \rho &= f_\theta(z, s) \quad \text{(normalised gene expression)} \\ x_g &\sim \text{NegBinomial}(l\rho_g, \theta_g) \end{aligned} \tag{21}$$

where s is a batch variable (which experiment produced the cell) and θ_g is the overdispersion parameter for gene g. The negative binomial distribution models the overdispersion of count data (variance $>$ mean) that the Poisson distribution cannot capture.

Optimal Transport for Trajectory Inference

Waddington-OT (Schiebinger et al., 2019) reconstructs cellular differentiation trajectories by solving an optimal transport problem between snapshots of cell state distributions at different times:

$$\pi_t^* = \arg\min_{\pi \in \Pi(\mu_{t_1}, \mu_{t_2})} \sum_{i,j} \pi_{ij} \|x_i^{t_1} - x_j^{t_2}\|^2 + \varepsilon H(\pi) \tag{22}$$

The optimal transport plan π_t^* specifies the probability that a cell in state x_i at time t_1 transitions to state x_j at time t_2. The entropy regularisation $\varepsilon H(\pi)$ (Sinkhorn) controls the sharpness of the coupling: small ε gives deterministic transport (one-to-one matching), large ε gives diffuse transport (spreading).

Cross-Reference to Other Parts. The OT-based trajectory inference connects directly to the flow matching framework of Part IV (Chapter 7): reconstructing a cellular differentiation trajectory is equivalent to learning the vector field that transports the distribution of undifferentiated cells to the distribution of differentiated cells. OT-FM (Part IV) with the scRNA-seq cell embeddings as x_0 (early time) and x_1 (late time) provides a principled generative model of differentiation.

Foundation Models for Genomics: Nucleotide Transformers and HyenaDNA

Historical and Motivational Context. *The success of BERT and GPT on protein sequences (ESM family) motivated analogous approaches for DNA sequences. The key difference: DNA has a much larger vocabulary challenge (4 nucleotides vs. 20 amino acids) but far longer sequences (billions of base pairs vs. hundreds of amino acids). Early models (DNABERT, Ji et al. 2021) used k-mer tokenisation and BERT-style pre-training on the human genome. HyenaDNA (Nguyen et al., 2023) addressed the length limitation using state-space model convolutions, enabling models trained on entire chromosomes (up to 10^6 nucleotides) rather than short windows.*

k-mer Tokenisation vs Byte-Pair Encoding for DNA

Standard BERT-style models tokenise DNA with overlapping k-mers: a sequence ATCGATCG with $k = 3$ produces tokens [ATC, TCG, CGA, GAT, ATC, TCG]. The vocabulary size is 4^k (64 for $k = 3$, 1024 for $k = 5$). The overlap creates redundancy: adjacent tokens share $k - 1$ nucleotides, providing implicit position information but limiting context length.

Byte-Pair Encoding (BPE) for DNA identifies frequent sub-sequences and replaces them with single tokens, building a vocabulary of common genomic motifs. This is analogous to the subword tokenisation used in language models and captures biological structure: common regulatory motifs (TATA box, CTCF binding sites) emerge as frequent tokens.

HyenaDNA: Single-Nucleotide Resolution at Genomic Scale

HyenaDNA (Nguyen et al., 2023) uses single-nucleotide tokenisation (vocabulary size $= 4$) and replaces attention with Hyena operators (long convolutions parameterised by implicit networks):

$$y_n = \sum_{k=0}^{L-1} h_k x_{n-k}, \quad h_k = \text{MLP}(k/L) \tag{23}$$

The convolution kernel h_k is parameterised by a small MLP evaluated at relative position k/L, enabling extremely long kernels ($L = 10^6$) without storing the kernel explicitly (only the MLP parameters are stored).

Training HyenaDNA on the human reference genome (single chromosomes at up to 10^6-nucleotide context) with next-nucleotide prediction achieves competitive performance on downstream tasks (variant effect prediction, gene expression prediction) while processing contexts $100\times$ longer than transformer-based models.

Evo: A Genomic Foundation Model Across All of Life

Evo (Nguyen et al., 2024) scales genomic foundation models to all domains of life: bacteria, archaea, eukaryotes, and viruses. Trained on a curated dataset of 2.7 million genomes at single-nucleotide resolution with a 131k token context window, Evo generates both coding and regulatory DNA de novo, designs novel CRISPR guide RNAs, and predicts the fitness effects of mutations in viral proteins—all without task-specific fine-tuning.

The key architectural choice: the StripedHyena architecture alternates Hyena layers (for long-range context) with attention layers (for precise token-level interaction), combining the complementary strengths of both.

Cross-Reference to Other Parts. Evo's approach of pre-training a generalist model on all available genomic sequences and then applying it to diverse tasks without fine-tuning is the genomics analogue of GPT-3's few-shot prompting (Part V, Chapter 6 on ICL): both leverage the information compressed into a large generalist model by conditioning on a small number of in-context examples. The HyenaDNA convolution is related to the state-space models used in Mamba (Chapter 5 this Part) and shares the same computational complexity $O(L \log L)$ via FFT.

Graph-Level Property Prediction and Molecular Fingerprinting

Graph Pooling: Global and Hierarchical

Graph-level prediction requires aggregating all node features into a single vector. Naive global sum/mean pooling: $h_G = \text{READOUT}(\{h_v^{(L)} : v \in G\})$ loses structural information—isomorphic graphs pool to the same vector, but non-isomorphic graphs may also pool to the same vector if nodes have the same multiset of representations.

DiffPool (Ying et al., 2018) learns to hierarchically cluster nodes: at each level ℓ, an assignment matrix $S^{(\ell)} \in \mathbb{R}^{n_\ell \times n_{\ell+1}}$ maps n_ℓ nodes to $n_{\ell+1}$ clusters: $H^{(\ell+1)} = (S^{(\ell)})^T H^{(\ell)}$,

$A^{(\ell+1)} = (S^{(\ell)})^T A^{(\ell)} S^{(\ell)}$. The assignment matrix is learned by a GNN, and is trained with an auxiliary link prediction loss $\|A^{(\ell)} - S^{(\ell)}(S^{(\ell)})^T\|_F^2$ that encourages clusters to correspond to dense subgraphs.

Weisfeiler-Lehman Graph Kernels

Before GNNs, graph kernels provided a principled approach to graph-level learning by defining a positive definite similarity function on graphs. The WL subtree kernel (Shervashidze et al., 2011) is based on the WL test:

1. Assign initial node labels $\ell_0(v) =$ atomic number.
2. At each iteration k: $\ell_k(v) = \text{hash}(\ell_{k-1}(v), \text{multiset}\{\ell_{k-1}(u) : u \in \mathcal{N}(v)\})$.
3. The feature vector $\phi_k(G) = [\text{count of each label at iteration } k]$.

The WL kernel: $K_{\text{WL}}(G_1, G_2) = \sum_k \langle \phi_k(G_1), \phi_k(G_2) \rangle$. GNNs with sum aggregation and injective hash functions (GIN) are equivalent to the WL test, connecting deep learning to the kernel literature.

Protein Engineering and Directed Evolution

Historical and Motivational Context. *Directed evolution (Nobel Prize, Arnold 2018) uses iterative cycles of mutation and selection to engineer proteins with desired properties. Classical directed evolution is expensive: each cycle requires expressing, purifying, and assaying thousands of protein variants. ML-guided directed evolution (Yang et al., 2019) replaces the assay with a machine learning fitness predictor, dramatically reducing the number of experimental evaluations needed to find high-fitness variants.*

Fitness Landscape Modelling

The fitness landscape assigns a scalar fitness $f(x) \in \mathbb{R}$ to each amino acid sequence $x \in \{1, \ldots, 20\}^L$. The landscape is *rugged* (many local optima) and *sparse* (only a tiny fraction of sequence space has been experimentally characterised).

Gaussian process regression on protein embeddings: $f(x) \sim \mathcal{GP}(m(x), k(x, x'))$ with a deep kernel $k(x, x') = k_{\text{base}}(\phi(x), \phi(x'))$ where ϕ is a protein language model embedding provides both fitness prediction and uncertainty quantification. The GP posterior provides: $f^*|x^*, \mathcal{D} \sim \mathcal{N}(\mu^*, \sigma^{*2})$ enabling Bayesian optimisation over sequence space.

Bayesian Optimisation for Protein Engineering

Bayesian optimisation (Snoek et al., 2012) selects the next sequence to evaluate by maximising an acquisition function that balances exploitation (high predicted fitness) and exploration (high predicted uncertainty):

Expected Improvement (EI): $\alpha_{\text{EI}}(x) = \mathbb{E}[\max(f(x) - f^*, 0)] = (\mu^* - f^*)\Phi(Z) + \sigma^* \phi(Z)$ where $Z = (\mu^* - f^*)/\sigma^*$, Φ is the normal CDF, ϕ is the normal PDF, and f^* is the current best observed fitness.

Upper Confidence Bound (UCB): $\alpha_{\text{UCB}}(x) = \mu^*(x) + \beta\sigma^*(x)$ where β controls the

exploration-exploitation tradeoff.

The key advantage of Bayesian optimisation over random mutation: it efficiently identifies high-fitness regions by maintaining a probabilistic model of the entire fitness landscape, rather than searching locally from a single current best.

Cross-Reference to Other Parts. Bayesian optimisation for protein engineering is the continuous analogue of the bandit algorithms of Part V (Chapter 8): both UCB acquisition and LinUCB select arms/sequences to evaluate by maximising an upper confidence bound, and both achieve near-optimal sample efficiency via the same optimism-in-the-face-of-uncertainty principle. The GP fitness model is the function-space Bayesian analogue of the value function in RL: both provide estimates of expected future rewards/fitness with uncertainty bounds that guide exploration.

Concluding Remarks on Part VII

Scientific AI represents the most mathematically demanding application of deep learning: architectures must respect the physical laws governing biological and chemical systems, expressed in the language of group theory, differential geometry, and statistical mechanics.

The AlphaFold2 success story is a masterclass in inductive bias encoding: every component of the architecture reflects a precise mathematical decision about what structure the problem has and how to encode it. AlphaFold3's extension to all biomolecular systems via diffusion, ESM-3's unification of sequence, structure, and function, and AlphaGenome's multi-scale genomic prediction all follow the same pattern: identify the relevant symmetry group, design an architecture that respects it exactly, and train at the scale where empirical data provides sufficient supervision.

The frontier of scientific AI is the *generative* direction: not just predicting known structures but designing new molecules with specified properties, generating new regulatory elements with specified expression patterns, and discovering new protein families with novel folds.

Appendix: Key Formulae Reference

A.1 Group Theory and Equivariance

SE(3) action: $(R,t)\cdot x = Rx+t, \quad (R_1,t_1)\circ(R_2,t_2) = (R_1R_2, R_1t_2+t_1)$

Schur's Lemma (equivariant maps): $L: V^{\ell_1} \to V^{\ell_2}$ equivariant $\Leftrightarrow$ ($\ell_1 = \ell_2$ and $L = \lambda I$) or $L = 0$

Clebsch-Gordan decomposition: $V^{\ell_1} \otimes V^{\ell_2} = \bigoplus_{\ell=|\ell_1-\ell_2|}^{\ell_1+\ell_2} V^{\ell}$

Spherical harmonic transform: $(R\cdot Y_\ell^m)(\hat{r}) = \sum_{m'} D^{(\ell)}_{m'm}(R) Y_\ell^{m'}(\hat{r})$

A.2 AlphaFold2 Key Operations

MSA row-wise attention with pair bias: $a_{ij} = \text{softmax}(Q_i K_j^T/\sqrt{d} + W_b z_{ij})$

Outer product mean: $z_{ij} \mathrel{+}= \text{mean}_s(m_{is} \otimes m_{js})$

IPA attention logit: $a_{ij} \propto \exp\left(\frac{1}{\sqrt{d}}[q_i^T k_j + b_{ij} - \frac{\gamma_q}{2}\sum_h \|\hat{q}_{ih} - \hat{k}_{jh}\|^2]\right)$

FAPE loss: $\mathcal{L}_{\text{FAPE}} = \frac{1}{NT}\sum_i \sum_p \|T_i^{-1} x_p - \hat{T}_i^{-1} x_p\|_2$

A.3 AlphaFold3 Diffusion

Forward process: $x_t = x_0 + \sigma_t \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, I)$

Training objective: $\mathcal{L}_{\text{diff}} = \mathbb{E}_{t,\varepsilon}[\lambda(t)\|D_\theta(x_0 + \sigma_t\varepsilon, t, s, z) - x_0\|^2]$

A.4 Equivariant GNNs

PaiNN equivariant update: $\Delta v_i = \sum_j \phi_v(s_i, s_j, d_{ij}) \cdot (r_j - r_i)/d_{ij}$

EGNN coordinate update: $x_i' = x_i + \sum_j (x_i - x_j)\phi_x(m_{ij})$

GIN aggregation: $h_v^{(k)} = \text{MLP}^{(k)}((1 + \varepsilon^{(k)})h_v^{(k-1)} + \sum_{u \in \mathcal{N}(v)} h_u^{(k-1)})$

A.5 AlphaGenome

Reverse complement symmetry: $\text{RC}(x_1 \cdots x_L) = c(x_L) \cdots c(x_1), \quad c(A) = T$, $c(T) = A$, $c(C) = G$, $c(G) = C$

PCGrad conflict resolution: $\hat{g}_t = g_t - (g_t \cdot g_{t'}/\|g_{t'}\|^2) g_{t'}$ if $g_t \cdot g_{t'} < 0$

MATHEMATICS OF DEEP LEARNING

PART VIII

AGENTIC AI, META-LEARNING, AND CONTINUAL LEARNING

Agentic LLMs · Multi-Agent RL · Inference-Time Compute
MAML · Prototypical Networks · Video Generation
Continual Learning · Topological Analysis of Training

Eight-Part Monograph Series on the Mathematical Principles of Neural Networks
Independent of Companion Volumes — Complete in Itself

Silvano Salvador

© Mathematics of Deep Learning Series

PART VIII

AGENTIC AI, META-LEARNING, AND CONTINUAL LEARNING

Agentic LLMs · Multi-Agent RL · Inference-Time Compute

MAML · Prototypical Networks · Video Generation

Continual Learning · Topological Analysis of Training

© Mathematics of Deep Learning Series

Silvano Salvador

AGENTIC AI, META-LEARNING, AND CONTINUAL LEARNING

Part VIII of the Mathematics of Deep Learning

Agentic LLMs · MARL · Inference-Time Compute · MAML · Video Generation · Continual Learning · Topological Analysis

Preface: Beyond Single-Step Prediction

The preceding parts of this treatise have addressed systems that produce a single output given a single input — a classification, a generated image, a predicted structure. The frontier of deep learning increasingly concerns systems that must act over extended time horizons, adapt to new tasks from few examples, maintain competence across a stream of changing

environments, and coordinate with other agents. These challenges require mathematical frameworks beyond standard supervised and generative learning: the theory of partially observable decision processes, the game-theoretic analysis of multi-agent systems, the meta-learning formulation of task distributions, and the continual learning framework for non-stationary distributions.

This part assembles the mathematics of these frontiers. Chapter 1 develops the POMDP framework for agentic LLMs and the game theory of multi-agent systems. Chapter 2 addresses inference-time compute scaling: MCTS, process reward models, and the GRPO algorithm. Chapter 3 extends diffusion and flow matching to video. Chapter 4 develops the theory of meta-learning with MAML as the central case. Chapter 5 addresses catastrophic forgetting and continual learning. Chapter 6 closes with the topological analysis of learning dynamics.

Chapter 1: Agentic LLMs and Multi-Agent Reinforcement Learning

Large language models deployed as autonomous agents — systems that perceive observations, execute tools, reason over multi-step plans, and interact with other agents — require a mathematical framework that extends beyond standard language modelling. The POMDP formalism provides this framework, and multi-agent extensions introduce game-theoretic considerations absent in single-agent settings.

1.1 The ReAct Framework as a POMDP

A **Partially Observable Markov Decision Process (POMDP)** is a tuple $(S, A, O, P, R, \Omega, \gamma)$ where S is the (hidden) state space, A the action space, O the observation space, $P: S\times A\to\Delta(S)$ the transition kernel, $R: S\times A\to\mathbb{R}$ the reward function, $\Omega: S\times A\to\Delta(O)$ the observation model, and γ the discount factor. The agent does not observe $s\in S$ directly but receives an observation $o\in O$ and must act based on its **belief state** — the posterior $b(s) = P(S=s \mid o_1,a_1,\ldots,o_t)$.

The ReAct framework (Yao et al., 2022) instantiates LLM agents as POMDPs where: the hidden state S is the world state (e.g., contents of a web page, database); observations O are textual tool outputs (search results, code

execution outputs); actions A are generated text strings (tool calls, reasoning steps, final answers); and the LLM's context window serves as an explicit **belief representation** — the concatenation of observations and actions up to time t constitutes the agent's belief state.

Theorem 1.1 (Context Window as Belief State)

For an LLM agent with context window $C = (o_1, a_1, \ldots, o_t, a_t)$, the conditional distribution $\pi_LLM(a_{t+1} \mid C)$ is a valid POMDP policy if and only if the context window is a sufficient statistic of the belief state — i.e., $P(s_t \mid C) = P(s_t \mid b(C))$. This holds under the Markov assumption on the world state but fails when (i) observations are informative about states outside the context window (long-horizon dependencies) or (ii) the context is truncated (finite context length). The failure mode is precisely long-horizon reasoning failure: the agent's policy cannot condition on relevant past observations that have been removed from the context, making it a suboptimal POMDP policy.

1.2 Tool Use and the Grounding Problem

LLM agent actions include **tool calls**: structured outputs that invoke external functions (web search, code execution, API calls) and receive structured responses. The **grounding problem** asks how to ensure that the LLM's internal representations of world state accurately reflect the actual state after tool execution.

Formally, let $T: A_tools \times S \to O \times S$ be the tool execution function. The agent policy must reason about $P(s_{t+1} \mid o_{t+1}, T(a_t, s_t))$ — the updated belief after observing the tool output. Grounding fails when the LLM hallucinates tool outputs inconsistent with T or when it ignores tool outputs in favour of its prior. The mathematical solution is **observation grounding**: explicitly inserting tool outputs into the context at the correct position and training the model to assign high probability to the correct next token given these observations, which requires specialised training on tool-augmented trajectories.

1.3 Multi-Agent Reinforcement Learning

In a multi-agent system, N agents with policies $\pi = (\pi_1,...,\pi_n)$ interact in a shared environment. The **joint policy** π induces a distribution over trajectories and each agent i has its own return $J_i(\pi)$. The three canonical cooperative/competitive settings are:

Cooperative: all agents share a reward, $R_1 = ... = R_n = R$. The team-optimal joint policy maximises $J(\pi) = \mathbb{E}[\Sigma_t \gamma^t R(s_t,a_t)]$.

Zero-sum: $\Sigma_i R_i = 0$ at every step. The solution concept is the Nash equilibrium.

General-sum: arbitrary reward functions. Both Nash equilibria and correlated equilibria are solution concepts.

Theorem 1.2 (Nash Equilibrium Existence)

In any finite N-player normal form game (finite action spaces, general-sum rewards), at least one Nash equilibrium in mixed strategies exists. Proof: the best-response correspondence BR_i(π_{-i}) = argmax_{πᵢ} Jᵢ(πᵢ, π_{-i}) is upper-hemicontinuous and convex-valued on the compact mixed strategy simplex. By Kakutani's fixed-point theorem, the joint best-response correspondence BR(π) = ×ᵢ BRᵢ(π_{-i}) has a fixed point π* = BR(π*) — which is precisely a Nash equilibrium. Note: the existence theorem does not imply uniqueness, computability in polynomial time, or convergence of RL algorithms to Nash equilibria.

1.4 QMIX and MAPPO: Cooperative MARL Algorithms

Independent Q-Learning (IQL) trains each agent's Q-function independently, ignoring other agents. This fails because from agent i's perspective, other agents are non-stationary — their policies change during training, violating the stationarity assumption required for Q-learning convergence.

QMIX (Rashid et al., 2018) addresses this with centralised training and decentralised execution: a **mixing network** $Q_{tot}(\tau, a) = f_{mix}(Q_1(\tau_1,a_1),...,Q_n(\tau_n,a_n))$ aggregates individual Q-values into a global value. The mixing network satisfies the **Individual-Global-Max (IGM) condition**:

$$\text{argmax}_a Q_{tot}(\tau,a) = (\text{argmax}_{a_1} Q_1(\tau_1,a_1), ..., \text{argmax}_{a_n} Q_n(\tau_n,a_n))$$

This requires $\partial Q_{tot}/\partial Q_i \geq 0$ (monotonicity), enforced by making all mixing network weights non-negative. Under IGM, each agent can act greedily with

respect to its own Q_i without coordination, yet the joint action is globally optimal for Q_{tot}.

MAPPO applies PPO to cooperative MARL with centralised value functions: the critic $V_\varphi(s)$ has access to the global state s (available during training) while each actor $\pi_i(a_i \mid o_i)$ uses only local observations. The centralised critic enables variance reduction via accurate baselines without violating the decentralised execution constraint.

1.5 Mean-Field Games and the Many-Agent Limit

For systems with $N \to \infty$ homogeneous agents, Mean-Field Game (MFG) theory (Lasry and Lions, 2007) provides a tractable approximation. In the mean-field limit, each agent interacts not with specific other agents but with the **population distribution** (mean field) $\mu_t \in \Delta(S)$ of all agents. The MFG equilibrium is a pair (π^*, μ^*) satisfying two coupled equations:

Forward equation: given policy π^*, the population evolves as $\mu_{t+1} = T(\mu_t, \pi^*)$

Backward equation: given distribution μ^*, the optimal policy is $\pi^* = \arg\max_\pi J_i(\pi, \mu^*)$

The fixed point (π^*, μ^*) exists under mild conditions (Lipschitz rewards, compact state/action spaces) and can be computed by iterating the forward-backward system. MFGs are tractable because the N^2-dimensional interaction space is compressed to the finite-dimensional space of distributions $\Delta(S)$. Applications include traffic routing, financial market modelling, and swarm behaviour.

Chapter 2: Inference-Time Compute Scaling

The Kaplan and Chinchilla scaling laws describe how model quality improves with training compute. A complementary scaling dimension — test-time compute — has emerged as equally important: allocating more compute at inference time (through tree search, iterative refinement, or sampling with verification) can recover the quality gap between smaller and larger models, with implications for the design of AI systems.

2.1 Monte Carlo Tree Search for LLMs

MCTS (Coulom, 2006) is a best-first tree search algorithm that balances exploration and exploitation via the UCB1 selection rule. Applied to LLM decoding, each tree node is a partial generation (token sequence), and each edge is a token extension. The UCB1 selection rule at node v selects child $c^*(v)$:

$$c^*(v) = \text{argmax}_c \left[Q(c)/N(c) + C\sqrt{(\log N(v)/N(c))} \right]$$

where $Q(c)$ is the total reward accumulated through node c, $N(c)$ its visit count, and C the exploration constant. The MCTS loop: **selection** (follow UCB1 to a leaf), **expansion** (generate candidate children by sampling from the LLM), **simulation** (roll out to completion with a rollout policy), **backpropagation** (update Q and N along the selected path).

Theorem 2.1 (UCB1 Regret Bound)

For a bandit problem with K arms and T rounds, UCB1 achieves expected cumulative regret: $\mathbb{E}[R_t] \leq \Sigma_i: \Delta_i>0 \, [8 \log(T)/\Delta_i + (1 + \pi^2/3)\Delta_i]$where $\Delta_i = \mu^* - \mu_i$ is the gap between the optimal arm mean μ^* and arm i. This is $O(\sqrt{(KT \log T)})$ total regret, matching the information-theoretic lower bound up to logarithmic factors. In the MCTS context, this means UCB1 exploration guarantees that the tree search converges to the optimal action at a rate governed by the reward gap Δ_i — actions close to optimal are explored often, clearly suboptimal actions are quickly abandoned.

2.2 Process Reward Models

Outcome Reward Models (ORMs) score complete solutions. **Process Reward Models (PRMs)** score each step of a reasoning chain, providing denser signal. A PRM assigns a score $r(x, s_1,\ldots,s_k)$ to the partial solution $(s_1,\ldots,s_k)$ given problem x, trained on human annotations of step correctness.

The PRM can be interpreted as estimating the **value function** of the reasoning process: $V_PRM(x, s_1,\ldots,s_k) \approx P(\text{correct final answer} \mid x, s_1,\ldots,s_k)$ — the probability that the current partial solution will lead to a correct answer. This connection to RL value functions is not merely analogical: the PRM satisfies a Bellman-like consistency condition:

$$V_PRM(x, s_1,\ldots,s_k) = \Sigma_\{s_{k+1}\} \, \pi(s_{k+1} \mid x, s_1,\ldots,s_k) \cdot V_PRM(x, s_1,\ldots,s_{k+1})$$

A perfect PRM would enable optimal MCTS: at each expansion step, rather than rolling out to completion, the PRM provides an immediate value estimate for each candidate extension, making MCTS computationally feasible for generation tasks where full rollouts are expensive.

2.3 Best-of-N with Verifier and Scaling Laws

Best-of-N sampling generates N candidate solutions, scores them with a verifier (ORM or PRM), and returns the highest-scoring solution. The accuracy of Best-of-N scales predictably with N:

$$P(\text{best-of-}N \text{ correct}) = 1 - (1-p)^N$$

where p is the per-sample accuracy. For $p = 0.1$ and $N = 100$, this gives $1 - 0.9^{100} \approx 0.99997$ — near-perfect accuracy from a model with 10% per-sample accuracy. The compute cost is $O(N)$ LLM evaluations plus $O(N)$ verifier evaluations. The scaling law $P \propto 1-(1-p)^N$ establishes that test-time compute scales **exponentially efficiently** for problems with independent errors: doubling N raises the error probability from $(1-p)^N$ to $(1-p)^{2N} = ((1-p)^N)^2$, squaring the remaining failure probability.

2.4 GRPO: Group Relative Policy Optimisation

GRPO (Shao et al., 2024) is the RL algorithm underlying DeepSeek-R1 and similar reasoning models. It replaces the PPO critic with a **group-relative baseline**: for each problem x, sample G responses $\{y_1,\ldots,y_G\}$ from the current policy and compute the normalised advantage:

$$\hat{A}_i = \frac{r(x_i,y_i) - \text{mean}(\{r(x_i,y_j)\}_j)}{\text{std}(\{r(x_i,y_j)\}_j)}$$

The GRPO objective is the PPO-clipped objective with this group-relative advantage:

$$\mathcal{L}_{GRPO}(\theta) = \mathbb{E}_{x,\{y_i\}\sim\pi_{\theta old}}\Big[\sum_i \min\big(r_i(\theta)\hat{A}_i,\ \text{clip}(r_i(\theta),1-\varepsilon,1+\varepsilon)\hat{A}_i\big) - \beta D_{KL}(\pi_\theta \| \pi_{ref})\Big]$$

where $r_i(\theta) = \pi_\theta(y_i|x)/\pi_{\theta old}(y_i|x)$. GRPO eliminates the value network entirely — saving ~50% of training memory and compute — while maintaining the variance reduction of advantage-based gradients via the within-group normalisation. The group-relative baseline is an unbiased estimator of the advantage under the policy: $\mathbb{E}_{y\sim\pi}[\hat{A}] = 0$ by construction.

Chapter 3: Video Generation and Spatiotemporal Models

Video generation extends image diffusion models to the temporal dimension, introducing new challenges: the 4D nature of video data (spatial × temporal), the long-range temporal dependencies between frames, the computational cost of attention over spatiotemporal sequences, and the physical consistency constraints that videos must satisfy.

3.1 Video as a 4D Spatiotemporal Signal

A video clip $v \in \mathbb{R}^{T\times H\times W\times C}$ (T frames, H×W pixels, C channels) lives in a much higher-dimensional space than a single image. The key challenges are:

Temporal consistency: adjacent frames must be coherent — objects must move smoothly, lighting must change gradually, physical laws must be respected.

Computational cost: full 3D self-attention has complexity $O((THW)^2)$. For $T=16$, $H=W=64$: sequence length 65,536 and attention cost 4.3 billion operations — prohibitive.

Temporal entanglement: models must learn both spatial appearance (what things look like) and temporal dynamics (how they move), two distinct aspects that ideally should be factored.

3.2 Factored Spatiotemporal Attention

Video Diffusion Models (Ho et al., 2022) address the computational challenge via **factored attention**: separate spatial attention (each frame attending within itself) and temporal attention (each spatial position attending across frames):

$$\text{Spatial: } a^{\text{spatial}}_{(t,i),(t,j)} = \text{softmax}(Q_{(t,i)} K_{(t,j)}^T/\sqrt{d}) \quad \text{[same frame } t\text{]}$$

$$\text{Temporal: } a^{\text{temporal}}_{(t,i),(t',i)} = \text{softmax}(Q_{(t,i)} K_{(t',i)}^T/\sqrt{d}) \quad \text{[same position } i\text{, different frames]}$$

Factored attention reduces complexity from $O((THW)^2)$ to $O(T(HW)^2 + (HW)T^2)$. For $T=16$, $H=W=64$: 2.7×10^8 — a 15× reduction. The factored approach loses some cross-frame-cross-spatial interactions, but empirically the independence

assumption between spatial and temporal processing is a good approximation for natural videos.

Theorem 3.1 (Expressive Power of Factored vs Full Attention)

Full spatiotemporal attention can represent any function of the video tensor v ∈ ℝ^{T×H×W×d}. Factored attention (spatial followed by temporal) can represent any function that factors as f(v) = g(h(v_{spatial}), v_{temporal}) where h aggregates spatial information and g combines spatial aggregates across time. The set of functions representable by factored attention is a strict subset of those representable by full attention: there exist video processing tasks (e.g., detecting diagonal spatiotemporal patterns) that require full attention. Empirically, natural video generation does not appear to require these extra-capacity functions, explaining why factored models achieve competitive quality.

3.3 3D RoPE and the DiT Architecture for Video

Sora (Brooks et al., 2024) is based on the **Diffusion Transformer (DiT)** architecture with a spatiotemporal extension. Videos are tokenised as $p_t \times p_H \times p_W$ spatiotemporal patches (e.g., 1×8×8 pixels per patch). Position encodings use **3D RoPE** — extending the 1D rotary position embedding of transformers to three dimensions by assigning separate frequency components to temporal, height, and width axes:

$$RoPE_3D(q_i, (t,h,w)) = R_t(t)\, R_h(h)\, R_w(w)\, q_i$$

where R_t, R_h, R_w are 2D rotation matrices in different subspaces of the embedding. 3D RoPE preserves the relative position property — the dot product $\langle R(t,h,w)q, R(t',h',w')k \rangle$ depends only on the relative offsets $(t-t', h-h', w-w')$ — crucial for generalisation to videos of different lengths and resolutions.

3.4 Flow Matching for Video Generation

Flow matching (Chapter 7 of Part IV) extends naturally to video: the vector field $v_\theta(x_t, t): \mathbb{R}^{T\times H\times W\times C} \times [0,1] \to \mathbb{R}^{T\times H\times W\times C}$ generates a flow from Gaussian noise to the video distribution. The conditional flow matching loss with optimal transport couplings:

$$\mathcal{L}_{CFM}(\theta) = \mathbb{E}_{t,x_0,x_1}[\| v_\theta(\varphi_t(x_0,x_1), t) - (x_1 - x_0) \|^2_F]$$

where $\|\cdot\|_F$ is the Frobenius norm over all spatiotemporal dimensions. The advantage over diffusion for video is that flow matching trajectories are straighter (by the OT coupling), requiring fewer sampling steps — critical for video where the high-dimensional output makes sampling expensive. CogVideoX and similar models use flow matching with 3D full attention, demonstrating high-quality text-to-video synthesis in 50 sampling steps.

Chapter 4: Meta-Learning — Learning to Learn

Meta-learning addresses the few-shot learning problem: given a task distribution, learn a prior that enables rapid adaptation to new tasks from very few examples. The mathematical framework distinguishes between the meta-learning phase (across many tasks) and the adaptation phase (within a single new task), and different algorithms make different assumptions about the structure of the adaptation.

4.1 The Meta-Learning Formulation

Let $p(\mathcal{T})$ be a distribution over tasks, each task $\mathcal{T}_i = (\mathcal{D}_i^{train}, \mathcal{D}_i^{test}, \mathcal{L}_i)$ consisting of a support set, query set, and loss function. The meta-learning objective is:

$$\min_\theta \mathbb{E}_{\mathcal{T}\sim p(\mathcal{T})}[\mathcal{L}_{\mathcal{T}}(f_{Adapt(\theta, \mathcal{D}^{train}_{\mathcal{T}})}; \mathcal{D}^{test}_{\mathcal{T}})]$$

The **Adapt** function maps the meta-parameters θ and task support data to task-specific parameters. Different meta-learning algorithms correspond to different parametric forms of Adapt:

MAML: gradient steps from θ on support data

ProtoNets: compute class prototypes from support embeddings

Meta-SGD: learn both initial parameters and per-parameter learning rates

4.2 MAML: Model-Agnostic Meta-Learning

MAML (Finn et al., 2017) sets $Adapt(\theta, \mathcal{D}^{train}) = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{D}^{train}}(f_\theta)$ — one (or K) gradient steps on the support set. The meta-objective becomes:

$$\min_\theta \Sigma_i \mathcal{L}_{\mathcal{T}_i}(f_{\theta - \alpha\nabla_\theta \mathcal{L}_{\mathcal{T}_i}^{train}(f_\theta)}) \quad \text{(meta-loss on query sets)}$$

The meta-gradient requires differentiating through the adaptation gradient step — computing the Hessian of the task loss with respect to θ:

$$\nabla_\theta \mathcal{L}_{\mathcal{T}_i}^{test}(\theta_i') = (I - \alpha\nabla^2_\theta \mathcal{L}_{\mathcal{T}_i}^{train}(\theta))\, \nabla_{\theta_i'} \mathcal{L}_{\mathcal{T}_i}^{test}(\theta_i')$$

Theorem 4.1 (MAML as Implicit Gradient Learning)

MAML with K gradient steps finds parameters θ such that, after K adaptation steps on any task $\mathcal{T}$ from p($\mathcal{T}$), the adapted parameters $\theta_{\mathcal{T}} = \theta - \alpha\Sigma_k\nabla\mathcal{L}$ are near the task optimum. The meta-gradient implicitly computes second-order information: it evaluates how a perturbation of θ propagates through K gradient steps to affect test performance. The MAML optimum θ* is characterised by: $\nabla_\theta \mathbb{E}_{\mathcal{T}}[\mathcal{L}_{\mathcal{T}}^{test}(\theta^*_{\mathcal{T}})] = 0$, where $\theta^*_{\mathcal{T}}$ is the result of K adaptation steps from θ*. This is a fixed-point condition: the meta-parameters are optimal if every task-adapted version is at or near its task optimum.

4.3 First-Order Approximations: Reptile and FOMAML

Computing the full MAML meta-gradient requires the Hessian of the task loss, which costs O(P²) memory. Two first-order approximations avoid this:

FOMAML ignores the second-order terms in the meta-gradient, using only $\nabla_{\theta_i'} \mathcal{L}_{\mathcal{T}_i}^{test}(\theta_i')$ as the meta-gradient. Empirically, FOMAML achieves 95%+ of full MAML performance at a fraction of the cost.

Reptile (Nichol et al., 2018) updates $\theta \leftarrow \theta + \varepsilon(\theta_i' - \theta)$ — moving θ toward the task-adapted parameters. This avoids any second-order computation. Reptile can be shown to approximate MAML in the first-order limit: it implicitly maximises within-task generalisation (the correlation between gradients on different minibatches of the same task), which is equivalent to MAML when the task distribution has a particular structure.

4.4 In-Context Learning vs MAML: Two Meta-Learning Paradigms

Large language models perform meta-learning via in-context learning (ICL): few-shot examples in the context serve as the support set, and the model's next-token prediction is the adapted prediction. The connection to MAML is:

ICL performs meta-learning through the forward pass (attention over context); MAML performs it through the backward pass (gradient descent on support data).

Akyürek et al. (2023) formalise this: a transformer trained on *(x,y)* pairs implements one step of gradient descent on the in-context examples via its attention mechanism, with *W_K^T W_Q* playing the role of the step size times the feature kernel. This makes transformer ICL a special case of kernel regression adaptation, with the context length playing the role of the support set size and the attention pattern implementing the kernel smoother.

Chapter 5: Continual and Lifelong Learning

Continual learning addresses the challenge of learning a sequence of tasks without forgetting previous ones — a fundamental limitation of standard neural network training, which exhibits catastrophic forgetting: gradient steps on new tasks overwrite the parameters relevant to old tasks. The mathematical framework connects Bayesian inference, the geometry of the loss landscape, and the information theory of parameter compression.

5.1 Catastrophic Forgetting: Geometric Analysis

Consider training on task $\mathscr{T}_1$ to convergence at θ_1^*, then training on task $\mathscr{T}_2$. The gradient update $\Delta\theta = -\alpha\nabla_\theta \mathscr{L}_{\mathscr{T}_2}(\theta)$ moves θ in a direction that minimises $\mathscr{L}_{\mathscr{T}_2}$ but may increase $\mathscr{L}_{\mathscr{T}_1}$. A second-order approximation around θ_1^* gives the change in task-1 loss:

$$\Delta L_{\mathscr{T}_1} \approx \Delta\theta^T \nabla^2_\theta \mathscr{L}_{\mathscr{T}_1}(\theta_1^*)\,\Delta\theta = \Delta\theta^T H_1 \Delta\theta$$

where $H_1 = \nabla^2 \mathscr{L}_{\mathscr{T}_1}$ is the Hessian of task 1's loss at θ_1^*. Forgetting is large when $\Delta\theta$ has large components along the high-curvature eigenvectors of H_1 — i.e., when the new task's gradient steps move θ in directions that matter for task 1.

5.2 Elastic Weight Consolidation (EWC)

EWC (Kirkpatrick et al., 2017) addresses forgetting by adding a regularisation term that penalises movement along dimensions important for previous tasks. The key insight is that the Fisher Information Matrix $F_1 = \mathbb{E}_{p_{data,1}}[\nabla_\theta \log p_\theta(y|x)\, \nabla_\theta \log p_\theta(y|x)^T]$ approximates the Hessian H_1 at θ_1^* (the equality $F = -\mathbb{E}[\nabla^2_\theta \log p_\theta]$ holds exactly; the approximation is $F \approx H$ at a maximum likelihood point).

The EWC objective for learning task $\mathscr{T}_2$ while retaining task $\mathscr{T}_1$ is:

$$\mathscr{L}_{EWC}(\theta) = \mathscr{L}_{\mathscr{T}_2}(\theta) + (\lambda/2)\, \Sigma_i\, F_{ii}(\theta_i - \theta_1^{*}{}_i)^2$$

where F_{ii} is the i-th diagonal element of the Fisher matrix. The quadratic penalty $F_{ii}(\theta_i - \theta_1^{*}{}_i)^2$ elastically anchors each parameter θ_i to its task-1 optimum with stiffness λF_{ii} — proportional to how important that parameter is for task 1 (high Fisher diagonal = high importance).

Theorem 5.1 (EWC as Laplace Approximation)

EWC minimises the negative log-posterior $p(\theta \mid \mathscr{D}_1, \mathscr{D}_2)$ where the posterior is approximated as: $\log p(\theta \mid \mathscr{D}_1, \mathscr{D}_2) \approx \log p(\theta \mid \mathscr{D}_2) + \log p(\theta \mid \mathscr{D}_1)$, and $\log p(\theta \mid \mathscr{D}_1)$ is approximated by the Laplace approximation around θ_1^*: $\log p(\theta \mid \mathscr{D}_1) \approx -\tfrac{1}{2}(\theta-\theta_1^*)^T F_1 (\theta-\theta_1^*)$. The EWC objective is the MAP estimator under this approximation. The approximation is exact when p_θ is an exponential family (Fisher = Hessian exactly) and approximate otherwise (curvature of the loss ≈ Fisher information at a maximum). The diagonal approximation of F_1 introduces further approximation error but reduces storage from $O(P^2)$ to $O(P)$.

5.3 Gradient Episodic Memory (GEM) and A-GEM

EWC approximates the constraint "do not increase task-1 loss" with a quadratic penalty. **GEM** (Lopez-Paz and Ranzato, 2017) enforces this constraint exactly. GEM stores an episodic memory $\mathscr{M}_t = \{(x_i, y_i)\}_i$ of examples from each past task and requires that gradient updates do not increase the loss on any stored example:

$$\min_{\Delta\theta} \|\Delta\theta - g\|^2 \quad \text{s.t.} \quad \langle \Delta\theta, g_{t'}, i\rangle \geq 0 \quad \forall (t', i): t' < t$$

where g is the current task's gradient and $g_{t',i}$ is the gradient on memory example (t', i). The constraint $\langle \Delta\theta, g_{t'} \rangle \geq 0$ ensures that the update does not increase the loss in the direction of past task gradients. The constrained QP is solved by projecting the current gradient g onto the constraint set — feasible if and only if the current and past gradients are not too conflicting. **A-GEM** (Chaudhry et al., 2019) approximates GEM by averaging memory gradients ($g_{ref} = \text{mean}_{t',i}[g_{t',i}]$) and projecting onto a single averaged constraint, reducing per-update cost from $O(|\mathscr{M}|P)$ to $O(P)$

5.4 Progressive Neural Networks and Synaptic Intelligence

Progressive Neural Networks (PNN, Rusu et al., 2016) allocate a new column of parameters for each new task and connect it to all previous columns via lateral connections. The previous columns are frozen, guaranteeing zero forgetting by construction. The lateral connections $h_{l i} = f_l(W_{l i} h_{l-1 i} + \Sigma_{j<i} U_{l j i}\, h_{l-1 j})$ allow the new task to leverage features learned by previous tasks (forward transfer) without modifying them (no forgetting). The cost is linear parameter growth: n tasks require n columns, limiting scalability.

Synaptic Intelligence (SI, Zenke et al., 2017) is an online EWC variant that accumulates importance weights ω_i during training (rather than computing the Fisher at convergence):

$$\omega_i = \Sigma_t\ (-\partial \mathscr{L} / \partial \theta_i \cdot \Delta\theta_i) \,/\, (\Delta\theta_i^2 + \xi)$$

The numerator measures the contribution of parameter i to the loss decrease during training; the denominator normalises by the total parameter change. SI provides online importance estimation, enabling continual learning in streaming settings where tasks do not have clear boundaries.

Chapter 6: Topological Analysis of Learning Dynamics

Topological data analysis provides tools to characterise the geometry of neural network representations and loss landscapes that complement spectral and statistical methods. This chapter applies persistent homology and related tools to understand learning dynamics, grokking, and the topology of feature spaces.

6.1 Simplicial Complexes and the Rips Complex

Given a point cloud $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^d$, the **Vietoris-Rips complex** at scale $\varepsilon > 0$ is the simplicial complex $VR_\varepsilon(X)$ containing all subsets $\sigma \subseteq X$ with diameter $\max_{x,y \in \sigma} d(x,y) \le \varepsilon$. The Rips complex is nested: $VR_\varepsilon \subseteq VR_{\varepsilon'}$ for $\varepsilon \le \varepsilon'$, forming a filtration. The homology groups $H_k(VR_\varepsilon(X))$ capture: H_0 = connected components, H_1 = loops, H_2 = enclosed voids.

6.2 Persistence Diagrams as Learning Metrics

The **persistence diagram** D_k of a filtration is a multiset of birth-death pairs $\{(b_i, d_i)\}$ encoding the topological features at each scale. The **bottleneck distance** between two persistence diagrams:

$$d_B(D, D') = \inf_{\gamma: D \to D'} \sup_{p \in D} \|p - \gamma(p)\|_\infty$$

Theorem 6.1 (Stability of Persistence Diagrams)

Let $f, g: X \to \mathbb{R}$ be two functions on a topological space X with $\|f - g\|_\infty \leq \varepsilon$. Then the bottleneck distance between their persistence diagrams satisfies $d_B(D(f), D(g)) \leq \varepsilon$. This stability theorem means that small perturbations of the function (e.g., small changes in network weights) produce small changes in the persistence diagram. Consequence for neural network analysis: persistence diagrams of activation functions or loss landscape sublevel sets vary continuously during training, enabling topological analysis of training dynamics that is robust to noise.

6.3 Topological Analysis of Loss Landscapes

The loss landscape $\mathscr{L}: \mathbb{R}^P \to \mathbb{R}$ can be analysed topologically via the sublevel set filtration $\{\mathscr{L} \leq c\}_{c \in \mathbb{R}}$. Connected components in H_0 correspond to disconnected minima basins; loops in H_1 correspond to "mountain passes" connecting basins. The **0-dimensional persistence** provides a rigorous measure of the number of distinct loss basins: a long-lived connected component (birth $b \approx \mathscr{L}_{min}$, death d large) is a deep, well-separated minimum; short-lived components are shallow local minima that merge quickly into deeper basins.

In overparameterised networks, topological analysis reveals that the loss landscape is **topologically trivial** at the global minimum: the set $\{\theta: \mathscr{L}(\theta) \leq \varepsilon\}$ for small ε is (approximately) connected, meaning that all global minima are accessible from each other via paths of near-zero loss. This is the **mode connectivity** phenomenon: linear or quadratic paths can connect different SGD solutions found from different random initialisations with near-zero loss along the entire path.

6.4 Topological Interpretation of Grokking

Grokking (delayed generalisation) can be characterised topologically. During the memorisation phase, the representation topology of training examples and

test examples are distinct: training examples form tight clusters (low H_0 persistence), while test examples are scattered. During the generalisation phase, the representation topology undergoes a phase transition: the representation space reorganises into a structure where training and test examples have similar topological signatures — the same connected components, the same loop structure.

Nanda et al.'s (2023) mechanistic account of grokking for modular arithmetic provides the precise topological picture: the **Fourier representation** $(\cos(2\pi ka/p), \sin(2\pi ka/p))$ corresponds to points on a circle in representation space. The H_1 persistent homology of the representation detects this circular structure: before grokking, H_1 is empty (no loops detected); after grokking, H_1 contains one long-lived loop corresponding to the circular Fourier structure. The grokking transition is precisely the moment this topological feature is born in the representation.

Concluding Remarks on Part VIII

The mathematical frameworks assembled in this part — POMDPs and multi-agent game theory, inference-time compute scaling, video diffusion, meta-learning, continual learning, and topological analysis — share a common theme: they address the problem of intelligence across time. Agentic systems must reason over extended horizons; meta-learning must leverage structure across a lifetime of tasks; continual learning must preserve knowledge across a stream of environments; topological analysis characterises the geometry of learning trajectories.

The connections to other parts of the treatise are numerous. GRPO and MAPPO are applications of the policy gradient theorem of Part V; EWC uses the Fisher information matrix of Part I; video flow matching applies the optimal transport framework of Part IV; the topological analysis of grokking connects to the algebraic topology of Part VI. Scientific AI (Part VII) increasingly incorporates meta-learning (few-shot adaptation to new protein families) and continual learning (updating models as new structural data arrives). The mathematical unity of the treatise reflects the mathematical unity of the discipline.

APPENDICES

Mathematical Notation · Theorems Reference · Proof Techniques · Index

Based on the work of Silvano Salvador

Appendix A: Mathematical Notation and Conventions

This appendix consolidates the notational conventions used throughout the eight parts of the treatise. The notation follows that of Salvador's Sezione 0, augmented as needed by standard conventions from the literature. Ambiguity is resolved in favour of clarity over brevity.

A.1 Sets, Spaces, and Structural Objects

Symbol	Meaning / Definition
$\mathbb{R}$, $\mathbb{C}$, $\mathbb{Z}$, $\mathbb{N}$	Real numbers, complex numbers, integers, natural numbers (0-inclusive)
$\mathbb{R}^n$, $\mathbb{R}^{m\times n}$	n-dimensional real vector space; space of real m×n matrices
$\mathbb{S}^{n-1}$	Unit (n-1)-sphere in $\mathbb{R}^n$: $\{x \in \mathbb{R}^n : \|x\|_2 = 1\}$
$\Delta(X)$	Set of probability distributions over X (probability simplex if X finite)
$\mathcal{H}$	Hilbert space; $\mathcal{G}$ denotes a general measurable space
$\mathcal{X}$, $\mathcal{Y}$	Input space, output space (generic)
$[n]$	The set {1, 2, ..., n}
$\mathcal{P}(X)$	Power set of X; $\mathcal{P}_k(X)$ = all k-element subsets
G, H	Groups (script letters); $g \in G$ denotes a group element
$GL(n)$, $O(n)$, $SO(n)$	General linear, orthogonal, special orthogonal groups over $\mathbb{R}$
$SE(3)$	Special Euclidean group in 3D: rigid motions (rotations + translations)
$\mathcal{G} = (V, E)$	Graph with vertex set V and edge set E

A.2 Vectors and Matrices

Symbol	Meaning
x, y, z	Column vectors (lower-case bold italic in equations)
x_i	i-th component of vector x; x_{ij} is (i,j) entry of matrix X
X, W, A, B	Matrices (upper-case); W^{(ℓ)} is the weight matrix at layer ℓ
X^{S}	Transpose of matrix X; X† (or X*) = conjugate transpose (Hermitian)
X^{-1}, X†	Matrix inverse; Moore-Penrose pseudoinverse
⊗	Kronecker product: (A ⊗ B)_{ij,kl} = A_{ik} B_{jl}
⊙	Hadamard (element-wise) product: (A ⊙ B)_{ij} = A_{ij} B_{ij}
diag(v)	Diagonal matrix with entries given by vector v
vec(X)	Column-major vectorisation of matrix X
Tr(X)	Trace of square matrix X = $\Sigma_i X_{ii}$
rank(X)	Rank (dimension of column space) of X
σ_i(X)	i-th singular value of X (σ_1 ≥ σ_2 ≥ ... ≥ 0)
‖X‖_F	Frobenius norm = (Σ_{ij} X^2_{ij})^{1/2} = (Σ_i σ_i^2)^{1/2}
‖X‖_*	Nuclear norm = Σ_i σ_i (sum of singular values)
$\|\mathbf{X}\|_2$	Spectral norm = σ_1(X) (largest singular value)
‖x‖_p	ℓ^p-norm = $(\Sigma_i \|x_i\|^p)^{1/p}$; ‖x‖_∞ = max_i $\|x_i\|$
⟨x, y⟩	Inner product = $x^T y$ (Euclidean); ⟨f, g⟩_ℋ for Hilbert space inner product

A.3 Calculus and Optimisation

Symbol	Meaning
∂f/∂x	Partial derivative of scalar f with respect to x
∇_θ f(θ)	Gradient vector of f at θ: $(\partial f/\partial\theta_1, ..., \partial f/\partial\theta_p)^T$
∇^2_θ f(θ)	Hessian matrix of f: H_{ij} = $\partial^2 f / \partial\theta_i \partial\theta_j$
J_f(x)	Jacobian of vector-valued map f: (J_f)_{ij} = ∂f_i/∂x_j

Symbol	Meaning
δL/δf	Functional derivative of functional L with respect to function f
∂_θ L	Shorthand for ∇_θ L (log-derivative notation)
d/dt, $\dot{\mathbf{x}}$	Time derivative of x(t)
∘	Function composition: (f ∘ g)(x) = f(g(x))
f*	Convex conjugate (Legendre-Fenchel transform): f*(y) = sup_x {⟨x,y⟩ - f(x)}
prox_{λf}(x)	Proximal operator: arg min_z {f(z) + (1/2λ)‖z - x‖²}
∂f(x)	Subdifferential of convex f at x (set of subgradients)
O(·), Θ(·), Ω(·)	Landau asymptotic notation (upper, tight, lower bounds)

A.4 Probability and Statistics

Symbol	Meaning
p(x), q(x)	Probability density or mass function (context determines which)
P(A)	Probability of event A
$\mathbb{E}$_p[f(X)]	Expectation of f(X) under distribution p; $\mathbb{E}$[·] when p is clear from context
Var_p[X]	Variance = $\mathbb{E}$[(X - $\mathbb{E}$X)²]
Cov(X,Y)	Covariance = $\mathbb{E}$[(X - $\mathbb{E}$X)(Y - $\mathbb{E}$Y)^T]
$\mathcal{N}$(μ, Σ)	Multivariate Gaussian with mean μ ∈ $\mathbb{R}^n$ and covariance Σ ∈ $\mathbb{R}$^{n×n}
$\mathcal{N}$(0, I_d)	Standard d-dimensional Gaussian
Cat(π)	Categorical distribution with probability vector π ∈ Δ^K
Ber(p)	Bernoulli distribution with success probability p
σ(x)	Logistic sigmoid: σ(x) = 1/(1 + e^{-x})
softmax(z)_i	Softmax: exp(z_i) / Σ_j exp(z_j)

Symbol	Meaning
lse(β, z)	Log-sum-exp: (1/β) log Σᵢ exp(β z_i)
D_KL(p ‖ q)	KL divergence: 𝔼_p[log p(X)/q(X)] ≥ 0; equals 0 iff p = q a.e.
H(p)	Entropy: -𝔼_p[log p(X)]; H(p,q) = -𝔼_p[log q(X)] = cross-entropy
I(X;Y)	Mutual information: H(X) - H(X∣Y) = D_KL(p(x,y) ‖ p(x)p(y))
W_p(μ, ν)	p-Wasserstein distance between probability measures μ, ν
D_TV(p, q)	Total variation distance: (1/2) Σ_x ∣p(x) - q(x)∣
x ~ p	'x is distributed according to p' / 'x is sampled from p'
x ⊥ y ∣ z	Conditional independence of x and y given z

A.5 Deep Learning–Specific Notation

Symbol	Meaning
L(θ), ℒ(θ)	Loss function (L) and ELBO / variational objective ($\mathscr{L}$)
f_θ	Neural network with parameters θ; f_θ: $\mathscr{X} \to \mathscr{Y}$
a^{(ℓ)}, z^{(ℓ)}	Post-activation and pre-activation at layer ℓ
δ^{(ℓ)}	Error signal at layer ℓ during backpropagation
W^{(ℓ)}, b^{(ℓ)}	Weight matrix and bias vector at layer ℓ
σ(·)	Activation function (context: logistic sigmoid or generic nonlinearity)
ReLU(x)	Rectified linear unit: max(0, x); component-wise for vectors
GELU(x)	Gaussian error linear unit: x · Φ(x) where Φ is Gaussian CDF
Q, K, V	Query, Key, Value matrices in attention mechanism
Attn(Q,K,V)	Scaled dot-product attention: softmax($QK^T/\sqrt{d}$) · V
G(θ)	Fisher Information Matrix (FIM): 𝔼[∇log p · ∇log p^T]
Θ(x,x')	Neural Tangent Kernel (NTK) evaluated at inputs x, x'
π_θ(a∣s)	Policy: probability of action a in state s with parameters θ

Symbol	Meaning
$V^\pi(s)$, $Q^\pi(s,a)$	Value function and action-value function under policy π
$\mathcal{T}^\pi$, $\mathcal{T}^*$	Bellman evaluation and optimality operators
r_t, G_t	Reward at time t; return (discounted sum of rewards from t)
$\mathcal{D}$	Dataset or replay buffer
ε_t	Reparametrisation noise $\varepsilon \sim \mathcal{N}(0,I)$ in VAE / diffusion
$s_\theta(x,t)$	Score network approximating $\nabla_x \log p_t(x)$

Appendix B: Compendium of Key Theorems

This appendix provides a self-contained statement of the principal theorems referenced across the eight parts of the treatise, organised by mathematical domain. For each theorem, the originating chapter and key references are given. Proofs are sketched where the technique is instructive; complete proofs are found in the cited sources.

B.1 Linear Algebra and Matrix Theory

Theorem B.1 (Singular Value Decomposition)

Every matrix $A \in \mathbb{R}^{m\times n}$ admits the factorisation $A = U\Sigma V^T$ where $U \in O(m)$, $V \in O(n)$, and $\Sigma = \mathrm{diag}(\sigma_1,\ldots,\sigma_r, 0,\ldots,0)$ with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$, $r = \mathrm{rank}(A)$.

The SVD is unique (up to sign) when the singular values are distinct.

Best rank-k approximation (Eckart-Young): $\|A - \hat{A}_k\|_F$ is minimised by $\hat{A}_k = U_k \Sigma_k V_k^T$ (first k columns of U, k×k top-left of Σ, first k columns of V). The error is $\|A - \hat{A}_k\|_F = (\Sigma_{i>k} \sigma_i^2)^{1/2}$.

Theorem B.2 (Marchenko-Pastur Law)

Let $X \in \mathbb{R}^{n\times p}$ have i.i.d. entries with mean 0 and variance σ^2/n; let $\gamma = p/n \to c \in (0,\infty)$.

The empirical spectral distribution of X^TX converges weakly a.s. to the Marchenko-Pastur law:

$$dF_c(\lambda) = \sqrt{(b-\lambda)(\lambda-a)} / (2\pi c\sigma^2\lambda) \cdot 1_{[a,b]}(\lambda) + \max(0, 1-1/c) \cdot \delta_0$$

where $a = \sigma^2(1 - \sqrt{c})^2$, $b = \sigma^2(1 + \sqrt{c})^2$.

Deviations from MP (heavy-tailed bulk, outlier eigenvalues) signal learned structure in trained networks.

Theorem B.3 (Weyl's Perturbation Theorem)

Let $A, E \in \mathbb{R}^{n\times n}$ be symmetric. Denote eigenvalues $\lambda_1(\cdot) \geq \ldots \geq \lambda_n(\cdot)$. Then:

$$\max_i |\lambda_i(A + E) - \lambda_i(A)| \leq \|E\|_2$$

Application: The Hessian $\nabla^2 L(\theta)$ of the loss is perturbed by mini-batch noise; the sharpness (largest Hessian eigenvalue) can change by at most the spectral norm of the noise. This bounds the rate of sharpness increase during training — a tool for analysing flat minimum methods (SAM).

B.2 Information Theory and Information Geometry

Theorem B.4 (Data Processing Inequality)

For any Markov chain $X \to Y \to Z$ (Y is a function of X, Z is a function of Y):

$I(X; Z) \le I(X; Y)$ and $I(X; Z) \le I(Y; Z)$

Mutual information cannot increase through processing. Consequence: no representation Z of input X can contain more information about label Y than X itself — the theoretical ceiling for learning is $I(X;Y)$.

Theorem B.5 (Fisher-Cramér-Rao Bound)

For any unbiased estimator $T(X_1,...,X_n)$ of a parameter $\theta \in \mathbb{R}^d$ from i.i.d. observations:

$\mathrm{Cov}(T) \ge G(\theta)^{-1} / n$ (in the Loewner / positive semidefinite sense)

where $G(\theta)$ is the Fisher Information Matrix. The bound is achieved by the MLE in regular families. Role in ML: the FIM is the metric on the statistical manifold; natural gradient descent is coordinate-invariant steepest descent in the FIM metric, not the Euclidean metric.

Theorem B.6 (Gibbs' Variational Principle)

For any distribution q and any measurable f:

$\log \mathbb{E}_p[\exp(f)] = \sup_q \{ \mathbb{E}_q[f] - D_{KL}(q \| p) \}$

The supremum is attained at $q^*(x) \propto p(x) \exp(f(x))$.

Consequences: (1) The ELBO identity in VAEs; (2) the optimal RLHF policy ($\pi^* \propto \pi_{ref} \cdot \exp(r/\beta)$); (3) the variational representation of the log-partition function in energy-based models.

B.3 Statistical Learning Theory

Theorem B.7 (PAC Learning Generalisation Bound via Rademacher Complexity)

Let $\mathscr{H}$ be a hypothesis class, $\mathscr{S} = \{(x_i, y_i)\}_{i=1}^m$ i.i.d. from D. With probability $\ge 1-\delta$:

for all $h \in \mathscr{H}$: $L_D(h) \le \hat{L}_S(h) + 2\mathfrak{R}_m(\mathscr{H}) + \sqrt{(\log(2/\delta) / 2m)}$

where $\mathbb{E}_m(\in) = _i _\sigma[\sup_{\{h_i\}} (1/m) \Sigma_i \sigma_i h(x_i)]$ is the empirical Rademacher complexity.

For neural networks with bounded weight norms, $\mathfrak{R}_m \le C \cdot \|W\|_F / \sqrt{m}$ (matrix spectral norms).

Theorem B.8 (Neural Tangent Kernel Convergence)

Consider a fully-connected network $f_\theta: \mathbb{R}^{n_0} \to \mathbb{R}$ with widths $n_1 = \ldots = n_L = n \to \infty$.
The NTK $\Theta_n(x,x') = \Sigma_p (\partial f_\theta(x)/\partial\theta_p)(\partial f_\theta(x')/\partial\theta_p)$ converges in probability to a deterministic kernel $\Theta_\infty(x,x')$ as $n \to \infty$. Under gradient flow on the squared loss, if the NTK Gram matrix $K = (\Theta_\infty(x_i,x_j))$
has minimum eigenvalue $\lambda_{min} > 0$, the training loss converges exponentially:

$$L(t) \le L(0) \cdot \exp(-2\lambda_{min} t)$$

B.4 Optimisation

Theorem B.9 (Convergence of SGD for Non-convex Functions)

Let $f: \mathbb{R}^d \to \mathbb{R}$ be L-smooth ($\|\nabla f(x) - \nabla f(y)\| \le L\|x-y\|$) and bounded below.
SGD with step size $\eta_t = \eta/\sqrt{t}$, where gradient estimates satisfy $\mathbb{E}[\|\hat{g}_t - \nabla f(\theta_t)\|^2] \le \sigma^2$:

$$(1/T)\ \Sigma_{t=1}^{T}\ \mathbb{E}[\|\nabla f(\theta_t)\|^2] \le O(\sigma / \sqrt{T}) + O(L\ \eta\ \sigma^2)$$

So after $T = O(\varepsilon^{-4})$ steps, expected gradient norm is $\le \varepsilon$ (first-order stationarity).
Adam achieves the same rate with per-coordinate learning rates, often with better constants.

Theorem B.10 (Mirror Descent and Natural Gradient as Bregman Projections)

Natural gradient descent is equivalent to Mirror Descent with the Bregman divergence
$D_F(\theta, \theta') = F(\theta) - F(\theta') - \langle \nabla F(\theta'), \theta - \theta' \rangle$ where F is the log-partition function:

$$\theta_{t+1} = \arg\min_\theta \{\ \eta\ \langle \nabla L(\theta_t), \theta \rangle + D_{KL}(p_\theta\ \|\ p_{\theta_t})\ \}$$

This makes explicit that natural gradient descent follows the steepest descent path in the Riemannian manifold of distributions, not in parameter space.

B.5 Generative Models

Theorem B.11 (ELBO Decomposition for VAEs)

For any distribution $q_\varphi(z|x)$ (approximate posterior) and model $p_\theta(x,z) = p(z)p_\theta(x|z)$:

$$\log p_\theta(x) = \mathcal{L}(\theta,\varphi;x) + D_{KL}(q_\varphi(z|x)\ \|\ p_\theta(z|x))$$

where $\mathscr{L} = \mathbb{E}_{q_\varphi}[\log p_\theta(x|z)] - D_{KL}(q_\varphi(z|x) \| p(z))$ is the ELBO.
Since $D_{KL} \geq 0$, the ELBO is a lower bound: $\log p_\theta(x) \geq \mathscr{L}(\theta,\varphi;x)$.
Maximising $\mathscr{L}$ jointly over θ, φ simultaneously maximises the log-likelihood lower bound and minimises the approximation gap between q_φ and the true posterior.

Theorem B.12 (Anderson Reverse-Time SDE)

Let $\{x_t\}_{t\in[0,T]}$ satisfy the forward SDE: $dx_t = f(x_t,t)dt + g(t)dW_t$
with marginal densities $p_t(x)$. The exact reverse-time SDE (Anderson 1982) is:

$$dx_t = [f(x_t,t) - g(t)^2 \nabla_x \log p_t(x_t)]\, dt + g(t)\, d\bar{W}_t$$

where $\bar{W}_t$ is a reverse-time Brownian motion.
The score function $\nabla_x \log p_t$ can be approximated by a neural network $s_\theta(x,t)$ trained via denoising score matching, converting learning the reverse SDE to supervised regression.

Theorem B.13 (Wasserstein GAN Duality)

By the Kantorovich-Rubinstein theorem, the 1-Wasserstein distance between p_{data} and p_θ is:

$$W_1(p_{data}, p_\theta) = \sup_{f:\, \mathrm{Lip}(f) \leq 1} \{ \mathbb{E}_{x\sim p_{data}}[f(x)] - \mathbb{E}_{x\sim p_\theta}[f(x)] \}$$

Wasserstein GANs optimise this by parameterising the 1-Lipschitz critic f with a neural network (weight clipping or gradient penalty to enforce the Lipschitz constraint), and training the generator to minimise W_1 rather than JS divergence. The advantage: W_1 is continuous even when supports are disjoint.

B.6 Reinforcement Learning

Theorem B.14 (Bellman Contraction and Value Iteration Convergence)

The Bellman optimality operator $\mathbb{R}^*: \to^S \mathbb{R} \circ ^S$ is a γ-contraction in $\|\cdot\|_\infty$:

$$\|\mathscr{T}^*V - \mathscr{T}^*U\|_\infty \leq \gamma \, \|V - U\|_\infty \quad \text{for all } V, U \in \mathbb{R}^S$$

By Banach's fixed-point theorem, $V^* = \lim_{k\to\infty} (\mathscr{T}^*)^k V_0$ for any V_0, with

$$\|(\mathscr{T}^*)^k V_0 - V^*\|_\infty \leq \gamma^k \, \|V_0 - V^*\|_\infty$$

Policy Iteration converges in finitely many steps (at most $|A|^{|S|}$ policies).

Theorem B.15 (DPO Equivalence to RLHF, Rafailov et al. 2023)

Under the Bradley-Terry preference model $p(y^+ \succ y^- | x) = \sigma(r^*(x,y^+) - r^*(x,y^-))$,
the policy that minimises $L_{DPO}(\theta) = -\mathbb{E}[\log \sigma(\beta \log \pi_\theta(y^+|x)/\pi_{ref}(y^+|x) - \beta \log \pi_\theta(y^-|x)/\pi_{ref}(y^-|x))]$
is identical to the optimal policy of the KL-constrained RL problem:

$$\pi^* = \arg\max_\pi \mathbb{E}[r^*(x,y)] - \beta D_{KL}(\pi \,\|\, \pi_{ref})$$

DPO thus implicitly fits a reward model and optimises the RL objective simultaneously, without requiring an explicit reward model or RL training loop.

B.7 Algebraic Topology and Statistical Physics

Theorem B.16 (Persistent Homology Stability)

Let $f, g: \mathcal{K} \to \mathbb{R}$ be functions on a simplicial complex $\mathcal{K}$. The bottleneck distance between their persistence diagrams satisfies:

$$d_B(PD(f), PD(g)) \leq \|f - g\|_\infty$$

Consequence: persistent homology is a stable topological summary. Small perturbations in the loss function cannot produce large changes in the topology of its sublevel filtration. This makes persistent homology a robust tool for comparing loss landscapes across training runs.

Theorem B.17 (Hopfield Network Capacity: Classical vs Modern)

Classical Hopfield (binary): $M_{max} = 0.138\, N$ random patterns retrievable
Modern Hopfield (continuous): $M_{max} = \exp(\alpha N)$ patterns for some $\alpha > 0$
The exponential increase stems from replacing the bilinear energy with a log-sum-exp energy, which is equivalent to computing attention scores. The update rule becomes $\sigma_{new} = \Xi\, \text{softmax}(\beta\, \Xi^T \sigma)$, identifying Modern Hopfield retrieval with Transformer attention.

Appendix C: Core Proof Techniques

This appendix catalogues the principal proof techniques that recur throughout the treatise, providing brief explanations and examples of their use. Mastery of these techniques is prerequisite to reading the research literature.

C.1 The Log-Derivative (Score Function) Trick

Whenever an expectation $\mathbb{E}_{p_\theta}[f(x)]$ must be differentiated with respect to θ, the log-derivative identity $\nabla_\theta p_\theta(x) = p_\theta(x) \nabla_\theta \log p_\theta(x)$ converts the derivative of the integral into an integral:

$$\nabla_\theta \mathbb{E}_{p_\theta}[f(x)] = \mathbb{E}_{p_\theta}[f(x) \nabla_\theta \log p_\theta(x)]$$

This enables Monte Carlo gradient estimation by sampling $x \sim p_\theta$. Uses: REINFORCE, variational inference BBVI, policy gradient theorem. The resulting estimator is unbiased but high-variance; control variates (baselines) reduce variance without introducing bias.

C.2 The Reparametrisation Trick

For continuous distributions where $x = g_\theta(\varepsilon)$ with $\varepsilon \sim p(\varepsilon)$ independent of θ (e.g., $x = \mu_\theta + \sigma_\theta \varepsilon$, $\varepsilon \sim \mathcal{N}(0,I)$), the gradient passes through the sampling operation:

$$\nabla_\theta \mathbb{E}_{p_\theta(x)}[f(x)] = \mathbb{E}_{p(\varepsilon)}[\nabla_\theta f(g_\theta(\varepsilon))]$$

The reparametrisation trick decouples the stochastic node from the parameter θ, enabling low-variance gradients. Uses: VAE, SAC, normalising flows, continuous diffusion. Fails for discrete distributions — the Gumbel-Softmax relaxation provides a biased but low-variance alternative.

C.3 McDiarmid's Inequality and Uniform Convergence

McDiarmid's inequality bounds the deviation of a function of independent random variables:

$$\text{If } f(X_1,\ldots,X_m) \text{ satisfies: } \sup_{x_i,x'_i} |f(\ldots,x_i,\ldots) - f(\ldots,x'_i,\ldots)| \le c_i$$

$$\text{then: } P(f(X) - \mathbb{E}[f(X)] \ge t) \le \exp(-2t^2 / \textstyle\sum_i c_i^2)$$

Uses: all PAC learning generalisation bounds. The function f is the empirical risk; the bounded difference condition holds with $c_i = 2/m$ for bounded losses. Combined with a union bound over $\mathcal{H}$ (or chaining for infinite $\mathcal{H}$), this yields uniform convergence bounds.

C.4 The Replica Method (Statistical Physics Heuristic)

The replica method computes quenched averages $\mathbb{E}_J[\log Z(J)]$ via the identity $\log Z = \lim_{n\to 0} \partial/\partial n\, \mathbb{E}[Z^n]$. For integer n, $Z^n = \Pi_{a=1}^{n} Z$ creates n independent copies ('replicas') that become coupled through the disorder average. After computing $\mathbb{E}[Z^n]$ analytically for integer n and identifying order parameters Q_{ab}, one analytically continues to $n \to 0$.

The method is non-rigorous (the $n \to 0$ limit lacks a general justification) but produces exact results that have subsequently been rigorously verified. Uses: Hopfield capacity, SK spin glass free energy, learning curve of linear classifiers, high-dimensional regression.

C.5 Stochastic Approximation and the ODE Method

Many RL and ML algorithms are stochastic approximation iterations: $\theta_{t+1} = \theta_t + \alpha_t\, h(\theta_t, \xi_t)$ where ξ_t is noise and h is the update direction. Convergence is established by showing that the 'mean ODE' $d\theta/dt = H(\theta) := \mathbb{E}[h(\theta, \xi)]$ has a globally attracting fixed point, and the stochastic iterates shadow this ODE (Ljung 1977, Borkar & Meyn 2000).

Key conditions: (1) Robbins-Monro step sizes $\Sigma\, \alpha_t = \infty$, $\Sigma\, \alpha_t^2 < \infty$; (2) h is bounded on compact sets; (3) the mean ODE is Lyapunov stable. Uses: Q-learning convergence (Watkins-Dayan), TD(λ) convergence, stochastic gradient methods.

Appendix D: Taxonomy of Neural Architectures and Inductive Biases

The following table organises major architectural families by their symmetry group (inductive bias), representative models, primary domain, and references to relevant chapters of the treatise.

Architecture Family	Symmetry / Inductive Bias	Representative Models	Domain	Treatise Part
MLP	None (dense)	FFN, MLP-Mixer	General	I
CNN	Translation equivariance	ResNet, EfficientNet, ConvNeXt	Vision	I, III
Transformer	Permutation equivariance + positional	GPT, BERT, ViT, Swin	Language, Vision	I, II
GNN	Graph isomorphism	GCN, GAT, MPNN, GraphTransformer	Graphs	II
LSTM/RNN	Sequential order	LSTM, GRU, Mamba, S4	Sequences	II
Equivariant Net	SE(3)/E(3) equivariance	SE(3)-TFN, EGNN, NequIP	Molecular/3D	I, II
VAE	Disentangled latent structure	β-VAE, NVAE, HiVAE	Generative	IV
GAN	Adversarial training	StyleGAN, BigGAN, CycleGAN	Generative	IV
NF	Invertibility + tractable Jacobian	RealNVP, Glow, NSF	Generative	IV
Diffusion	Reverse SDE	DDPM, LDM, DiT, SiT	Generative	IV
EBM	Energy function	EBM, JEM, Modern Hopfield	Generative/AM	IV, VI
RL Policy	Markov, temporal structure	PPO, SAC, TD3, AlphaZero	Control	V

Appendix E: Glossary of Technical Terms

A concise glossary of technical terms used throughout the treatise, with brief formal definitions.

Attention. A weighted sum of values, where weights are computed as a softmax of dot products between queries and keys. The core operation of Transformer models.

Backpropagation. Efficient computation of gradients in a computational graph via the chain rule, traversed in reverse topological order (backward mode autodiff).

Bayesian Inference. Updating a prior distribution $p(\theta)$ to a posterior $p(\theta \mid \text{data}) \propto p(\text{data} \mid \theta)p(\theta)$ using Bayes' theorem; used in BNNs and Laplace approximation.

Contrastive Loss. A loss that pushes together representations of matched pairs and pushes apart unmatched pairs; InfoNCE is the mutual-information-maximising variant.

Diffusion Model. A generative model that learns to reverse a Gaussian noising process via score matching or denoising score matching.

ELBO. Evidence Lower BOund: a tractable lower bound on $\log p(x)$ used in VAE and variational inference; ELBO = $\mathbb{E}_q[\log p(x \mid z)] - D_{KL}(q \| p)$.

Emergent Capability. A capability absent in small models and present in large models, appearing discontinuously as scale increases.

Equivariance. A map f is equivariant to group G if $f(gx) = gf(x)$ for all $g \in G$. Invariance is the special case $f(gx) = f(x)$.

Fisher Information Matrix. $G(\theta) = \mathbb{E}[\nabla \log p \cdot \nabla \log p^T]$; the Riemannian metric on the statistical manifold; curvature of the KL divergence to second order.

Flow Matching. A generative model that learns a vector field $v_\theta(x,t)$ whose ODE $dx/dt = v_\theta$ transports a source distribution to the data distribution.

Grokking. The phenomenon where a network memorises training data but later (after extended training) discovers a generalising algorithm.

Hilbert Space. A complete inner product space; RKHS is the Hilbert space of functions associated to a kernel, central to kernel methods and SVMs.

Importance Sampling. Estimating $\mathbb{E}_p[f]$ by sampling from q: $\mathbb{E}_p[f] = \mathbb{E}_q[f \cdot p/q]$. Used in off-policy RL (importance ratio π/π_{old}).

Inductive Bias. A constraint built into model architecture that makes certain hypotheses more likely, typically reflecting domain knowledge.

KL Divergence. $D_{KL}(p \| q) = \mathbb{E}_p[\log(p/q)] \geq 0$; measures 'extra bits' needed to encode p using code optimised for q; asymmetric.

LoRA. Low-Rank Adaptation: fine-tuning by learning $\Delta W = BA$ with $r \ll \min(m,n)$; reduces trainable parameters from mn to $r(m+n)$.

MCMC. Markov Chain Monte Carlo: algorithms (MH, HMC, NUTS) for sampling from unnormalised distributions by constructing ergodic Markov chains.

Mutual Information. $I(X;Y) = H(X) - H(X \mid Y) = D_{KL}(p(x,y) \| p(x)p(y))$; measures statistical dependence; central to information bottleneck and contrastive learning.

Natural Gradient. $\nabla^{nat}_\theta L = G(\theta)^{-1} \nabla_\theta L$; steepest descent in the Fisher-Rao metric; invariant under reparametrisation of θ.

NTK. Neural Tangent Kernel: the limiting kernel of an infinite-width network, constant during gradient flow training (lazy training regime).

Optimal Transport. $W_p(\mu,\nu)$: the cost of the cheapest way to transport mass from μ to ν; the p-Wasserstein distance; used in GANs, flow matching, and alignment.

Persistent Homology. Topological summary of a filtration of simplicial complexes; records birth and death of connected components, loops, and voids as scale varies.

Rademacher Complexity. $\mathfrak{R}_m(\mathscr{H})$: a data-dependent measure of hypothesis class capacity that tightly bounds generalisation; based on correlation with random sign patterns.

Replica Method. A statistical physics heuristic for computing $\mathbb{E}[\log Z]$ via analytic continuation from integer n (replicas) to $n \to 0$.

Residual Connection. $x_{\ell+1} = x_\ell + F(x_\ell)$; additive skip connection that stabilises gradient flow by providing a gradient highway of slope 1.

Score Function. $\nabla_x \log p(x)$; the gradient of the log-density; learned by score-matching and used to drive the reverse diffusion SDE.

Superposition. The hypothesis that neural networks represent more features than neurons by storing them as near-orthogonal directions in activation space.

VC Dimension. The largest set of points that can be shattered (all 2^n labellings achieved) by some hypothesis; measures expressiveness of a hypothesis class.

Wasserstein Distance. $W_p(\mu,\nu) = (\inf_{\gamma \in \Gamma(\mu,\nu)} \int c^p \, d\gamma)^{1/p}$; a proper metric on probability distributions that metrises weak convergence.

BIBLIOGRAPHY

Based on the work of Silvano Salvador

Bibliographic Note

This bibliography provides the complete references for claims, theorems, and empirical results cited across all parts of the treatise. Entries are organised thematically, mirroring the structure of the treatise itself, with a final section for the foundational source volumes by Silvano Salvador. Within each section, entries are ordered roughly chronologically to convey the intellectual history of each subfield.

Citation tags use the first three letters of the first author's surname followed by a two-digit year (e.g., [Vap95] for Vapnik 1995). In case of collision, a letter suffix disambiguates (e.g., [Sch17a] vs [Sch17b]). Preprints on arXiv are cited by their arXiv identifier where journal publication was not available at the time of writing.

I. Foundations: Linear Algebra, Optimisation, and Automatic Differentiation

Linear Algebra and Matrix Theory

[GVL13] G.H. Golub, C.F. Van Loan. *Matrix Computations, 4th ed.*. Johns Hopkins University Press, 2013. Standard reference for numerical linear algebra, SVD, and eigendecompositions.

[Bha97] R. Bhatia. *Matrix Analysis*. Springer Graduate Texts in Mathematics, 1997. Comprehensive treatment of spectral theory, perturbation bounds, and matrix functions.

[Wey12] H. Weyl. *Das asymptotische Verteilungsgesetz der Eigenwerte linearer partieller Differentialgleichungen*. Mathematische Annalen, 71(4), 1912. Classical perturbation bound for eigenvalues; modern statement in Bhatia (1997).

[Mar67] V.A. Marchenko, L.A. Pastur. *Distribution of eigenvalues for some sets of random matrices*. Mathematics of the USSR-Sbornik, 1(4), 1967. Foundational RMT result for empirical spectral distribution of sample covariance matrices.

[Mar18] C.H. Martin, M.W. Mahoney. *Implicit Self-Regularization in Deep Neural Networks*. arXiv:1810.01075, 2018. Heavy-tail self-regularisation hypothesis; power-law spectral analysis of trained networks.

[Voi85] D. Voiculescu. *Symmetries of some reduced free product C*-algebras*. Springer LNM 1132, 1985. Introduction to free probability theory; S-transform multiplicativity.

[And10] G.W. Anderson, A. Guionnet, O. Zeitouni. *An Introduction to Random Matrices*. Cambridge University Press, 2010. Standard graduate textbook on RMT, Wigner semicircle, MP law, free probability.

Backpropagation and Automatic Differentiation

[Rum86] D.E. Rumelhart, G.E. Hinton, R.J. Williams. *Learning representations by back-propagating errors*. Nature, 323, 1986. Classic statement of backpropagation for multilayer networks.

[Bay88] Y. Le Cun. *A theoretical framework for back-propagation*. Proceedings of the 1988 Connectionist Models Summer School, 1988. Automatic differentiation perspective on backpropagation.

[Bay15] A.G. Baydin, B.A. Pearlmutter, A.A. Radul, J.M. Siskind. *Automatic Differentiation in Machine Learning: a Survey*. JMLR, 18(153), 2018. Comprehensive survey of forward and reverse mode AD, and their ML applications.

[Pas19] A. Paszke et al.. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. NeurIPS 2019, 2019. PyTorch autodiff engine; dynamic computation graph and eager execution.

Optimisation

[Kin15] D.P. Kingma, J. Ba. *Adam: A Method for Stochastic Optimization.* ICLR 2015, 2015. Adam optimizer; adaptive moment estimation; bias correction.

[Los17] I. Loshchilov, F. Hutter. *Decoupled Weight Decay Regularization.* ICLR 2019, 2017. AdamW; shows that standard L2 and Adam weight decay are not equivalent.

[Bot10] L. Bottou. *Large-Scale Machine Learning with Stochastic Gradient Descent.* COMPSTAT 2010, 2010. Theoretical foundations of SGD for large-scale ML; learning rate schedules.

[Nes83] Y. Nesterov. *A method for solving the convex programming problem with convergence rate O(1/k²).* Soviet Mathematics Doklady, 27(2), 1983. Nesterov momentum; accelerated gradient descent for convex objectives.

[Mar20] D. Masters, C. Luschi. *Revisiting Small Batch Training for Deep Neural Networks.* arXiv:1804.07612, 2018. Empirical study of batch size effects on training dynamics and generalisation.

[For18] P. Foret, A. Kleiner, H. Mobahi, B. Neyshabur. *Sharpness-Aware Minimization for Efficiently Improving Generalization.* ICLR 2021, 2021. SAM: seeking flat minima by minimising the worst-case loss in a parameter neighbourhood.

[Ama98] S. Amari. *Natural Gradient Works Efficiently in Learning.* Neural Computation, 10(2), 1998. Natural gradient descent; Fisher information metric; invariance under reparametrisation.

[Mar15] J. Martens, R. Grosse. *Optimizing Neural Networks with Kronecker-factored Approximate Curvature.* ICML 2015, 2015. K-FAC: Kronecker-factored approximation of the Fisher information matrix.

II. Architectures and Geometric Inductive Biases

Transformer Architecture

[Vas17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin. *Attention Is All You Need*. NeurIPS 2017, 2017. Original Transformer architecture; multi-head self-attention; encoder-decoder.

[Dev19] J. Devlin, M.W. Chang, K. Lee, K. Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. NAACL 2019, 2019. Masked language modelling pre-training; bidirectional attention.

[Bro20] T.B. Brown et al.. *Language Models are Few-Shot Learners*. NeurIPS 2020, 2020. GPT-3 (175B); few-shot and zero-shot in-context learning; scaling to 175B parameters.

[Dao22] T. Dao, D. Fu, S. Ermon, A. Rudra, C. Ré. *FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness*. NeurIPS 2022, 2022. IO-aware tiled attention; reduces memory from $O(S^2)$ to $O(S)$; enables long contexts.

[Dao23] T. Dao. *FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning*. ICLR 2024, 2023. Improved work distribution and parallelism for modern GPU architectures.

[Su24] J. Su, Y. Lu, S. Pan, A. Murtadha, B. Wen, Y. Liu. *RoFormer: Enhanced Transformer with Rotary Position Embedding*. Neurocomputing, 2024. RoPE: rotary position embeddings; multiplicative position encoding compatible with linear attention.

[Kat23] A. Katharopoulos, A. Vyas, N. Pappas, F. Fleuret. *Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention*. ICML 2020, 2020. Linear attention via kernel feature maps; reduces attention complexity from $O(S^2)$ to $O(S)$.

[Mit23] D. De, S. Smith, A. Fernando et al.. *Griffin: Mixing Gated Linear Recurrences with Local Attention*. arXiv:2402.19427, 2024. Hybrid linear-recurrence plus local attention; efficient long-sequence modelling.

Equivariant and Geometric Architectures

[Coh16] T.S. Cohen, M. Welling. *Group Equivariant Convolutional Networks*. ICML 2016, 2016. G-CNNs; equivariance to discrete symmetry groups; group convolution theorem.

[Tho18] N. Thomas et al.. *Tensor Field Networks: Rotation- and Translation-Equivariant Neural Networks for 3D Point Clouds*. arXiv:1802.08219, 2018. SE(3)-equivariant networks via spherical harmonic tensor products.

[Jum21] J. Jumper et al.. *Highly accurate protein structure prediction with AlphaFold*. Nature, 596, 2021. AlphaFold 2; IPA mechanism; FAPE loss; SE(3)-equivariance in protein structure prediction.

[Bro22] J.L. Batzner et al.. *E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials.* Nature Communications, 13, 2022. NequIP; SE(3)-equivariant GNNs for molecular dynamics force fields.

Graph Neural Networks

[Kip17] T.N. Kipf, M. Welling. *Semi-Supervised Classification with Graph Convolutional Networks.* ICLR 2017, 2017. GCN; spectral graph convolution; Chebyshev polynomial approximation.

[Vel18] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio. *Graph Attention Networks.* ICLR 2018, 2018. GAT; attention-weighted aggregation over graph neighbourhoods.

[Xu19] K. Xu, W. Hu, J. Leskovec, S. Jegelka. *How Powerful are Graph Neural Networks?.* ICLR 2019, 2019. WL-test equivalence of MPNNs; GIN architecture achieving maximum WL-expressiveness.

[Bod22] C. Bodnar, F. Di Giovanni, B. Chamberlain, P. Liò, M. Bronstein. *Neural Sheaf Diffusion: A Topological Perspective on Heterophily in GNNs.* NeurIPS 2022, 2022. Cellular sheaves on graphs; sheaf Laplacian; sheaf neural networks.

Sequential and State Space Models

[Hoc97] S. Hochreiter, J. Schmidhuber. *Long Short-Term Memory.* Neural Computation, 9(8), 1997. LSTM; gated cell state; gradient highways through forget gate.

[Gu22] A. Gu, K. Goel, C. Ré. *Efficiently Modeling Long Sequences with Structured State Spaces.* ICLR 2022, 2022. S4; HiPPO matrices; FFT-based SSM convolution; O(L log L) computation.

[Gu23] A. Gu, T. Dao. *Mamba: Linear-Time Sequence Modeling with Selective State Spaces.* arXiv:2312.00752, 2023. Selective SSM; content-dependent B,C matrices; hardware-aware parallel scan.

[Pen23] B. Peng et al.. *RWKV: Reinventing RNNs for the Transformer Era.* EMNLP 2023, 2023. RWKV; linear attention as recurrence; WKV operator.

[Pol23] M. Poli et al.. *Hyena Hierarchy: Towards Larger Convolutional Language Models.* ICML 2023, 2023. Hyena; subquadratic long convolutions; implicit parameterisation.

III. Statistical Learning Theory and Probabilistic Methods

Learning Theory

[Vap95] V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995. VC dimension, SRM, SVM; PAC learning framework.

[Bar02] P.L. Bartlett, S. Mendelson. *Rademacher and Gaussian Complexities: Risk Bounds and Structural Results*. JMLR, 3, 2002. Rademacher complexity; data-dependent generalisation bounds; tighter than VC.

[Ney15] B. Neyshabur, R. Tomioka, N. Srebro. *Norm-Based Capacity Control in Neural Networks*. COLT 2015, 2015. Spectral norm bounds for neural networks; parameter norm vs capacity.

[Bel20] M. Belkin, D. Hsu, S. Ma, S. Mandal. *Reconciling modern machine-learning practice and the classical bias-variance tradeoff*. PNAS, 116(32), 2019. Double descent; benign overfitting; interpolation threshold.

[Jac18] A. Jacot, F. Gabriel, C. Hongler. *Neural Tangent Kernel: Convergence and Generalization in Neural Networks*. NeurIPS 2018, 2018. NTK; infinite-width convergence; lazy training regime.

[Aro19] S. Arora, S. Du, W. Hu, Z. Li, R. Wang. *Fine-Grained Analysis of Optimization and Generalization for Overparameterized Two-Layer Neural Networks*. ICML 2019, 2019. NTK convergence guarantees for two-layer networks; overparameterised regime.

Bayesian Methods and MCMC

[Nea93] R.M. Neal. *Probabilistic Inference Using Markov Chain Monte Carlo Methods*. Technical Report CRG-TR-93-1, University of Toronto, 1993. Foundational reference for HMC in ML; MCMC for neural networks.

[Hof14] M.D. Hoffman, A. Gelman. *The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo*. JMLR, 15(1), 2014. NUTS; adaptive HMC; No-U-Turn criterion; basis for Stan's sampler.

[Wel11] M. Welling, Y.W. Teh. *Bayesian Learning via Stochastic Gradient Langevin Dynamics*. ICML 2011, 2011. SGLD; stochastic gradient MCMC; injecting gradient noise for posterior sampling.

[Mac03] D.J.C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003. Comprehensive Bayesian ML reference; Laplace approximation; free energy.

Variational Inference and Uncertainty Quantification

[Kin14] D.P. Kingma, M. Welling. *Auto-Encoding Variational Bayes.* ICLR 2014, 2014. VAE; ELBO; reparametrisation trick; amortised variational inference.

[Res15] D.J. Rezende, S. Mohamed. *Variational Inference with Normalizing Flows.* ICML 2015, 2015. Normalising flows for flexible approximate posteriors; planar/radial flows.

[Bur16] Y. Burda, R. Grosse, R. Salakhutdinov. *Importance Weighted Autoencoders.* ICLR 2016, 2016. IWAE; tighter ELBO via importance-weighted samples; gap to log p(x).

[Ang19] A. Angelopoulos, S. Bates. *A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification.* arXiv:2107.07511, 2021. Conformal prediction; distribution-free coverage guarantees for any predictor.

[Lac17] B. Lakshminarayanan, A. Pritzel, C. Blundell. *Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles.* NeurIPS 2017, 2017. Deep ensembles; calibrated uncertainty via ensemble disagreement.

IV. Generative Models

GANs

[Goo14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio. *Generative Adversarial Networks.* NeurIPS 2014, 2014. Original GAN; minimax game; JS divergence minimisation.

[Arj17] M. Arjovsky, S. Chintala, L. Bottou. *Wasserstein Generative Adversarial Networks.* ICML 2017, 2017. WGAN; Kantorovich-Rubinstein duality; gradient penalty for Lipschitz constraint.

[Kar20] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, T. Aila. *Analyzing and Improving the Image Quality of StyleGAN.* CVPR 2020, 2020. StyleGAN2; style-based generator; path length regularisation; FID records.

Normalising Flows

[Die14] L. Dinh, D. Krueger, Y. Bengio. *NICE: Non-linear Independent Components Estimation.* ICLR Workshop 2015, 2014. NICE; additive coupling layers; first tractable normalising flow.

[Die17] L. Dinh, J. Sohl-Dickstein, S. Bengio. *Density estimation using Real-valued Non-Volume Preserving (Real NVP) transformations.* ICLR 2017, 2017. RealNVP; affine coupling layers; bijective generative model.

[Kin18] D.P. Kingma, P. Dhariwal. *Glow: Generative Flow with Invertible 1×1 Convolutions.* NeurIPS 2018, 2018. Glow; learned permutation via invertible 1×1 convolutions; high-quality image synthesis.

[Dur19] C. Durkan, A. Bekasov, I. Murray, G. Papamakarios. *Neural Spline Flows.* NeurIPS 2019, 2019. NSF; monotone rational-quadratic splines as coupling layers; expressive normalising flow.

Score-Based Models and Diffusion

[Son19] Y. Song, S. Ermon. *Generative Modeling by Estimating Gradients of the Data Distribution.* NeurIPS 2019, 2019. Score matching with Langevin dynamics; annealed IS for sample quality.

[Ho20] J. Ho, A. Jain, P. Abbeel. *Denoising Diffusion Probabilistic Models.* NeurIPS 2020, 2020. DDPM; Gaussian diffusion; denoising score matching as variational lower bound.

[Son21] Y. Song, J. Sohl-Dickstein, D.P. Kingma, A. Kumar, S. Ermon, B. Poole. *Score-Based Generative Modeling through Stochastic Differential Equations.* ICLR 2021, 2021. Unification of SMLD and DDPM via SDEs; VE/VP/subVP SDEs; probability flow ODE.

[Son22] J. Song, C. Meng, S. Ermon. *Denoising Diffusion Implicit Models*. ICLR 2021, 2021. DDIM; deterministic sampling from diffusion models; 10–50× inference speedup.

[Lip22] W. Lipman, R. Chen, H. Ben-Hamu, M. Nickel, M. Le. *Flow Matching for Generative Modeling*. ICLR 2023, 2022. Flow Matching; conditional vector fields; OT-FM for straight flows.

[Rom22] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, B. Ommer. *High-Resolution Image Synthesis with Latent Diffusion Models*. CVPR 2022, 2022. LDM / Stable Diffusion; diffusion in VQ-regularised latent space; cross-attention conditioning.

[Son23] Y. Song, P. Dhariwal, M. Chen, I. Sutskever. *Consistency Models*. ICML 2023, 2023. Consistency models; self-consistency property; direct one-step generation.

Self-Supervised and Contrastive Learning

[Che20] T. Chen, S. Kornblith, M. Norouzi, G. Hinton. *A Simple Framework for Contrastive Learning of Visual Representations*. ICML 2020, 2020. SimCLR; NT-Xent loss; data augmentation for contrastive self-supervised learning.

[Gri20] J.B. Grill et al.. *Bootstrap Your Own Latent: A New Approach to Self-Supervised Learning*. NeurIPS 2020, 2020. BYOL; self-distillation without negative pairs; online/target network asymmetry.

[Bar21] J. Zbontar, L. Jing, I. Misra, Y. LeCun, S. Deny. *Barlow Twins: Self-Supervised Learning via Redundancy Reduction*. ICML 2021, 2021. Barlow Twins; cross-correlation matrix towards identity; information redundancy.

Energy-Based Models and Hopfield

[Ram21] H. Ramsauer, B. Schäfl, J. Lehner, P. Seidl, M. Widrich, L. Gruber, M. Holzleitner, M. Pavlović, G.K. Sandve, V. Greiff, D. Kreil, M. Kopp, G. Klambauer, J. Brandstetter, S. Hochreiter. *Hopfield Networks is All You Need*. ICLR 2021, 2021. Modern Hopfield networks; exponential capacity; equivalence with attention.

[Lec06] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, F. Huang. *A Tutorial on Energy-Based Learning*. Predicting Structured Data, MIT Press, 2006. EBMs; energy function learning; contrastive divergence.

V. Reinforcement Learning and Alignment

MDP Theory and Value-Based Methods

[Put94] M.L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, 1994. Standard MDP reference; existence and uniqueness of optimal policy; PI convergence.

[Wat92] C.J.C.H. Watkins, P. Dayan. *Q-learning*. Machine Learning, 8(3-4), 1992. Q-learning algorithm; convergence proof via stochastic approximation theory.

[Mni15] V. Mnih et al.. *Human-level control through deep reinforcement learning*. Nature, 518, 2015. DQN; experience replay; target network; Atari benchmark.

[Bel17] M.G. Bellemare, W. Dabney, R. Munos. *A Distributional Perspective on Reinforcement Learning*. ICML 2017, 2017. Distributional RL; C51; Wasserstein contraction; return distribution.

Policy Gradient Methods

[Wil92] R.J. Williams. *Simple statistical gradient-following algorithms for connectionist reinforcement learning*. Machine Learning, 8(3-4), 1992. REINFORCE; score function gradient estimator; baseline variance reduction.

[Sut00] R.S. Sutton, D.A. McAllester, S.P. Singh, Y. Mansour. *Policy Gradient Methods for Reinforcement Learning with Function Approximation*. NeurIPS 2000, 2000. Policy Gradient Theorem; actor-critic; compatible function approximation.

[Sch16] J. Schulman, P. Moritz, S. Levine, M. Jordan, P. Abbeel. *High-Dimensional Continuous Control Using Generalized Advantage Estimation*. ICLR 2016, 2016. GAE; bias-variance tradeoff in advantage estimation; λ interpolation.

[Sch15] J. Schulman, S. Levine, P. Abbeel, M. Jordan, P. Moritz. *Trust Region Policy Optimization*. ICML 2015, 2015. TRPO; KL-constrained policy update; Fisher-vector products via conjugate gradient.

[Sch17] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov. *Proximal Policy Optimization Algorithms*. arXiv:1707.06347, 2017. PPO; clipped importance ratio surrogate; simpler and more robust than TRPO.

MaxEnt RL and Continuous Control

[Haa18] T. Haarnoja, A. Zhou, P. Abbeel, S. Levine. *Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor*. ICML 2018, 2018. SAC; soft Bellman equations; entropy-regularised RL; automatic temperature tuning.

[Fuj18] S. Fujimoto, H. van Hoof, D. Meger. *Addressing Function Approximation Error in Actor-Critic Methods.* ICML 2018, 2018. TD3; clipped double Q; target policy smoothing; delayed policy updates.

[Lil16] T.P. Lillicrap et al.. *Continuous control with deep reinforcement learning.* ICLR 2016, 2016. DDPG; deterministic policy gradient; actor-critic for continuous action spaces.

Model-Based RL

[Jan19] M. Janner, J. Fu, M. Zhang, S. Levine. *When to Trust Your Model: Model-Based Policy Optimization.* NeurIPS 2019, 2019. MBPO; theoretical bound on model error; short model rollouts for safe data augmentation.

[Haf19] D. Hafner, T. Lillicrap, J. Ba, M. Norouzi. *Dream to Control: Learning Behaviors by Latent Imagination.* ICLR 2020, 2019. Dreamer; RSSM; latent imagination; policy optimisation via imagined trajectories.

[Haf23] D. Hafner, J. Lillicrap, P. Norouzi, T.P. Lillicrap. *Mastering Diverse Domains through World Models.* arXiv:2301.04104, 2023. DreamerV3; universal hyperparameters; symlog predictions; free-bits KL.

[Sil18] D. Silver et al.. *A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play.* Science, 362, 2018. AlphaZero; MCTS + neural network; self-play without human knowledge.

[Sch20] J. Schrittwieser et al.. *Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model.* Nature, 588, 2020. MuZero; latent dynamics model; model-based planning without observation reconstruction.

RLHF, Alignment, and Preference Learning

[Chr17] P.F. Christiano, J. Leike, T.B. Brown, M. Martic, S. Legg, D. Amodei. *Deep Reinforcement Learning from Human Preferences.* NeurIPS 2017, 2017. RLHF; preference learning from human comparisons; reward model training.

[Sti20] N. Stiennon et al.. *Learning to summarize from human feedback.* NeurIPS 2020, 2020. RLHF for summarisation; PPO + reward model; first large-scale NLP application.

[Ouy22] L. Ouyang et al.. *Training language models to follow instructions with human feedback.* NeurIPS 2022, 2022. InstructGPT; RLHF pipeline for instruction following; PPO-based fine-tuning.

[Raf23] R. Rafailov, A. Sharma, E. Mitchell, C.D. Manning, S. Ermon, C. Finn. *Direct Preference Optimization: Your Language Model is Secretly a Reward Model.* NeurIPS 2023, 2023. DPO; reward reparametrisation; elimination of explicit RL stage from RLHF.

[Eth24] K. Ethayarajh, W. Xu, N. Muennighoff, J. Jurafsky, D. Kiela. *KTO: Model Alignment as Prospect Theoretic Optimization.* ICML 2024, 2024. KTO; Kahneman-Tversky prospect theory loss; binary feedback without preference pairs.

[Men24] Y. Meng, M. Xia, D. Chen. *SimPO: Simple Preference Optimization with a Reference-Free Reward.* NeurIPS 2024, 2024. SimPO; length-normalised reward; margin-based loss; reference-free alignment.

[Bai22] Y. Bai et al.. *Constitutional AI: Harmlessness from AI Feedback.* arXiv:2212.08073, 2022. Constitutional AI; RLAIF; critique-revision self-improvement; scalable oversight.

[Gao22] L. Gao, J. Schulman, J. Hilton. *Scaling Laws for Reward Model Overoptimization.* arXiv:2210.10760, 2022. KL-return Pareto frontier; Goodhart's law in RLHF; reward hacking dynamics.

Emergent Capabilities and In-Context Learning

[Wei22] J. Wei et al.. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models.* NeurIPS 2022, 2022. CoT prompting; intermediate reasoning steps; arithmetic, commonsense, symbolic tasks.

[Wei23] J. Wei, Y. Tay, R. Bommasani et al.. *Emergent Abilities of Large Language Models.* TMLR 2022, 2022. Operational definition of emergence; phase transitions in capability with scale.

[Xie22] S.M. Xie, A. Raghunathan, P. Liang, T. Ma. *An Explanation of In-context Learning as Implicit Bayesian Inference.* ICLR 2022, 2022. ICL as Bayesian inference over latent concepts; posterior concentration with examples.

[Aky22] E. Akyürek, D. Schuurmans, J. Andreas, T. Ma, D. Zhou. *What learning algorithm is in-context learning? Investigations with linear models.* ICLR 2023, 2022. ICL as gradient descent; Transformer layers implement one step of ridge regression.

[Fen23] N. Feng, Q. Zhou, C. Gu, H. He. *Towards Revealing the Mystery behind Chain of Thought: a Theoretical Perspective.* NeurIPS 2023, 2023. CoT and circuit complexity; TC^0 vs NC^1; depth-multiplying effect of chain of thought.

[Lew20] P. Lewis et al.. *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks.* NeurIPS 2020, 2020. RAG; dense retrieval + generation; open-domain QA.

Mechanistic Interpretability

[Ola20] C. Olah, N. Cammarata, L. Schubert, G. Goh, M. Petrov, S. Carter. *Zoom In: An Introduction to Circuits.* Distill, 2020. Circuits framework; curve detectors; multimodal neurons; circuit analysis in CNNs.

[Ols22] K. Olsson, N. Elhage, N. Nanda, N. Joseph, N. DasSarma et al.. *In-context Learning and Induction Heads.* Transformer Circuits Thread, 2022. Induction heads; two-layer circuit; phase transition during training.

[Elh22] N. Elhage, T. Henighan, A. Joseph, N. Nanda, N. Olsson et al.. *Toy Models of Superposition*. Transformer Circuits Thread, 2022. Superposition hypothesis; polytope geometry of feature representation; phase transitions.

[Cun23] H. Cunningham, A. Ewart, L. Riggs, R. Huben, L. Sharkey. *Sparse Autoencoders Find Highly Interpretable Features in Language Models*. arXiv:2309.08600, 2023. SAEs for mechanistic interpretability; monosemantic feature discovery.

VI. Advanced Mathematical Structures

Statistical Physics of Learning

[Hop82] J.J. Hopfield. *Neural networks and physical systems with emergent collective computational abilities.* PNAS, 79(8), 1982. Classical Hopfield network; energy function; memory capacity and retrieval.

[Ama87] D.J. Amit, H. Gutfreund, H. Sompolinsky. *Statistical Mechanics of Neural Networks near Saturation.* Annals of Physics, 173(1), 1987. Replica calculation of Hopfield capacity; AGS critical load $\alpha_c \approx 0.138$.

[Mez87] M. Mézard, G. Parisi, M. Virasoro. *Spin Glass Theory and Beyond.* World Scientific, 1987. Replica method; Parisi RSB; SK model exact solution; spin glass theory.

[Eng01] A. Engel, C. Van den Broeck. *Statistical Mechanics of Learning.* Cambridge University Press, 2001. Statistical physics approach to ML; storage capacity; generalisation in perceptrons.

[Bor22] B. Bordelon, C. Pehlevan. *Spectral Bias Outside the Neural Tangent Kernel Regime.* ICLR 2022, 2022. Spectral learning curves; power-law scaling from data covariance spectrum.

Algebraic Topology for ML

[Ede08] H. Edelsbrunner, J. Harer. *Persistent Homology — a Survey.* Contemp. Mathematics 453, 2008. Persistent homology; filtrations; persistence diagrams; Betti numbers.

[Coh07] D. Cohen-Steiner, H. Edelsbrunner, J. Harer. *Stability of Persistence Diagrams.* Discrete & Computational Geometry, 37(1), 2007. Bottleneck stability theorem; robustness of persistent homology to perturbations.

[Car09] G. Carlsson. *Topology and Data.* Bulletin of the AMS, 46(2), 2009. TDA survey; mapper algorithm; applications to real data.

[Han20] J. Hansen, R. Ghrist. *Opinion Dynamics on Discourse Sheaves.* SIAM J. Applied Mathematics, 81(5), 2020. Cellular sheaves on graphs; sheaf Laplacian; connection Laplacian.

[Pow22] A. Power, Y. Gur-Ari, D. Bahri, R. Anderson, J. Steinhardt. *Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets.* arXiv:2201.02177, 2022. Grokking phenomenon; delayed generalisation; algorithmic datasets.

Causal Inference and OOD Generalisation

[Pea09] J. Pearl. *Causality: Models, Reasoning, and Inference, 2nd ed..* Cambridge University Press, 2009. SCM; do-calculus; interventional and counterfactual distributions.

[Pet17] J. Peters, D. Janzing, B. Schölkopf. *Elements of Causal Inference*. MIT Press, 2017. Causal models; identifiability; ICM principle; invariant causal prediction.

[Arj19] M. Arjovsky, L. Bottou, I. Gulrajani, D. Lopez-Paz. *Invariant Risk Minimization*. arXiv:1907.02893, 2019. IRM; invariant predictors across environments; causal feature selection.

[Sag20] S. Sagawa, P.W. Koh, T.B. Hashimoto, P. Liang. *Distributionally Robust Neural Networks for Group Shifts*. ICLR 2020, 2020. Group DRO; worst-group accuracy; underrepresented group robustness.

Scaling Laws and Emergence

[Kap20] J. Kaplan, S. McCandlish, T. Henighan et al.. *Scaling Laws for Neural Language Models*. arXiv:2001.08361, 2020. Power-law scaling in N, D, C; compute-optimal allocation; chinchilla precursor.

[Hof22] J. Hoffmann et al.. *Training Compute-Optimal Large Language Models*. NeurIPS 2022, 2022. Chinchilla scaling law; equal scaling of N and D; 20 tokens per parameter.

[Sch23] R. Schaeffer, B. Miranda, S. Koyejo. *Are Emergent Abilities of Large Language Models a Mirage?*. NeurIPS 2023, 2023. Emergence as metric artefact; smooth improvement under linear metrics.

Large-Scale Fine-Tuning and Distributed Training

[Hu21] E.J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen. *LoRA: Low-Rank Adaptation of Large Language Models*. ICLR 2022, 2021. LoRA; low-rank weight perturbation; efficient fine-tuning of LLMs.

[Det23] T. Dettmers, A. Pagnoni, A. Holtzman, L. Zettlemoyer. *QLoRA: Efficient Finetuning of Quantized LLMs*. NeurIPS 2023, 2023. QLoRA; 4-bit NF4 quantisation; double quantisation; 65B on single GPU.

[Raj20] S. Rajbhandari, J. Rasley, O. Ruwase, Y. He. *ZeRO: Memory Optimizations Toward Training Trillion Parameter Models*. SC 2020, 2020. ZeRO stages 1/2/3; parameter, gradient, and optimizer state sharding.

[Zha23] Y. Zhao, A. Gu, R. Varma, L. Luo et al.. *PyTorch FSDP: Experiences on Scaling Fully Sharded Data Parallel*. VLDB 2023, 2023. FSDP; ZeRO-3 in PyTorch; per-unit parameter sharding; forward/backward gather.

[Che16] T. Chen, B. Xu, C. Zhang, C. Guestrin. *Training Deep Nets with Sublinear Memory Cost*. arXiv:1604.06174, 2016. Gradient checkpointing; √L memory-compute tradeoff; activation recomputation.

Multimodal Models

[Rad21] A. Radford, J.W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, I. Sutskever. *Learning Transferable Visual Models from Natural Language Supervision*. ICML 2021, 2021. CLIP; contrastive image-text pre-training; zero-shot transfer; InfoNCE loss.

[Ala22] J.B. Alayrac et al.. *Flamingo: a Visual Language Model for Few-Shot Learning*. NeurIPS 2022, 2022. Flamingo; Perceiver Resampler; cross-attention VLM; few-shot multimodal learning.

[Liu23] H. Liu, C. Li, Q. Wu, Y.J. Lee. *Visual Instruction Tuning*. NeurIPS 2023, 2023. LLaVA; linear projector from CLIP to LLM; multimodal instruction tuning.

[Cut13] M. Cuturi. *Sinkhorn Distances: Lightspeed Computation of Optimal Transport Distances*. NeurIPS 2013, 2013. Entropic regularised OT; Sinkhorn algorithm; efficient W distance computation.

VII. Source Volumes: The Salvador Cycle

This treatise is a synthesis and commentary on the following eleven volumes by Silvano Salvador, which constitute the primary source material. Each volume is identified by its title and the sections it covers from the Salvador numbering system.

Volume	Title	Coverage	Topics
Matematica delle reti neurali (Volume 1)	Linear Algebra for Neural Networks	Sections §1–§90	Vectors, matrices, eigendecompositions, SVD, dimensionality reduction.
Matematica delle reti neurali (Volume 2)	Calculus, Backpropagation, and Optimisation	Sections §91–§200	Gradient descent, backpropagation, Adam, loss landscapes, normalisation.
Matematica delle reti neurali (Volume 3)	Statistical Learning Theory	Sections §201–§280	VC dimension, Rademacher complexity, PAC learning, NTK, double descent.
Matematica delle reti neurali (Volume 4)	Information Geometry	Sections §281–§360	Fisher metric, natural gradient, K-FAC, information bottleneck, RKHS.
Matematica delle reti neurali (Volume 5)	Architectures and Equivariance	Sections §361–§450	Transformer, attention variants, CNN equivariance, GNN, WL-test.
Matematica delle reti neurali (Volume 6)	Generative Models I	Sections §451–§530	VAE, GAN, normalising flows, score matching, DDPM, DDIM.
Matematica delle reti neurali (Volume 7)	Generative Models II and Diffusion	Sections §531–§600	SDE diffusion, consistency models, flow matching, LDM, EBM, Hopfield.

Volume	Title	Coverage	Topics
Matematica delle reti neurali (Volume 8)	Reinforcement Learning	Sections §601–§660	MDP, Bellman, policy gradients, SAC, TD3, MBRL, RLHF, DPO.
Matematica delle reti neurali (Volume 9)	Bayesian and Probabilistic Methods	Sections §661–§720	MCMC, HMC, NUTS, SGLD, VI, ELBO, BBVI, IWAE, BNN, conformal prediction.
Matematica delle reti neurali (Volume 10)	Advanced Mathematics	Sections §721–§790	Spin glasses, replicas, free probability, algebraic topology, causal inference.
Matematica delle reti neurali (Volume 11)	Scaling, Fine-Tuning, and Multimodal	Sections §791–§860	Scaling laws, emergence, LoRA, QLoRA, ZeRO, FSDP, CLIP, VLMs.

The section numbering §1–§860 is the unified cross-volume numbering used throughout the present treatise to pinpoint specific results within the Salvador cycle.

Appendix B: Technical Supplement for Part VII — Scientific AI and Geometric Deep Learning

This appendix collects detailed proofs, derivations, and reference tables that support the main text of Part VII. All notation follows Appendix A. Equation numbers of the form (B.x) refer exclusively to this appendix.

B.1 Detailed Proof of IPA SE(3) Invariance (Theorem 2.2 of Part VII)

We provide a self-contained proof that the Invariant Point Attention (IPA) attention weights are invariant under any global rigid transformation $g = (R, t) \in SE(3)$, following Jumper et al. (2021) with minor notational adjustments.

Setup. Let $T_i = (R_i, t_i) \in SE(3)$ be the rigid frame assigned to residue i, so that the global rotation R and translation t act by $T_i \mapsto g \circ T_i = (RR_i, Rt_i + t)$. For each attention head h, residue i has query points $\tilde{q}_{ih} \in \mathbb{R}^3$ expressed in the local frame of residue i, and key points $\tilde{k}_{jh} \in \mathbb{R}^3$ expressed in the local frame of residue j. Their global-frame positions are:

$$p_{ih} = R_i \tilde{q}_{ih} + t_i, \quad k_{jh} = R_j \tilde{k}_{jh} + t_j.$$

Step 1: Distance under g. After the global transformation $g = (R, t)$, the new global positions are $p'_{ih} = R(R_i \tilde{q}_{ih}) + (Rt_i + t)$ and similarly for k'_{jh}. The squared distance between the transformed query and key points is:

$$\| p'_{ih} - k'_{jh} \|^2 = \| R(R_i \tilde{q}_{ih} - R_j \tilde{k}_{jh}) \|^2 = \| R_i \tilde{q}_{ih} - R_j \tilde{k}_{jh} \|^2 = \| p_{ih} - k_{jh} \|^2$$

where the second equality uses the fact that $R \in SO(3)$ is an isometry ($\|Rv\| = \|v\|$ for all v). The translational term cancels because both p' and k' receive the same translation $Rt_i + t$ and $Rt_j + t$ respectively, whose difference is $R(t_i - t_j)$, which is absorbed into the rotation-invariant norm.

Step 2: Invariance of attention logits. The IPA attention logit between residues i and j is:

$$a_{ij} = (1/\sqrt{(d_h + \gamma m)}) \, [q_i^T k_j + (\gamma m/2) \, \Sigma_h \, \| p_{ih} - k_{jh} \|^2]$$

The first term $q_i^T k_j$ is a dot product between sequence-space features and is independent of the 3D frames entirely. The second term is a sum of squared distances in global frame, which Step 1 shows is invariant under g. Therefore a_{ij} is SE(3)-invariant, and so are the softmax weights $A_{ij} = \mathrm{softmax}_j(a_{ij})$. ■

B.2 Clebsch-Gordan Decomposition for $\ell \leq 2$

The following table gives the irreducible decomposition of the tensor product $V_{\ell_1}^1 \otimes V_{\ell_2}^2$ for angular momentum values $\ell_1, \ell_2 \le 2$, relevant to the equivariant message-passing architectures NequIP, MACE, and SEGNN. The triangle rule $|\ell_1 - \ell_2| \le \ell \le \ell_1 + \ell_2$ determines the output types.

$\ell_1 = 0, \ell_2 = 0$: $V^0 \otimes V^0 = V^0$ (scalar ⊗ scalar = scalar)

$\ell_1 = 0, \ell_2 = 1$: $V^0 \otimes V^1 = V^1$ (scalar ⊗ vector = vector)

$\ell_1 = 1, \ell_2 = 1$: $V^1 \otimes V^1 = V^0 \oplus V^1 \oplus V^2$ ($3 \otimes 3 = 1 \oplus 3 \oplus 5$)

$\ell_1 = 1, \ell_2 = 2$: $V^1 \otimes V^2 = V^1 \oplus V^2 \oplus V^3$ ($3 \otimes 5 = 3 \oplus 5 \oplus 7$)

$\ell_1 = 2, \ell_2 = 2$: $V^2 \otimes V^2 = V^0 \oplus V^1 \oplus V^2 \oplus V^3 \oplus V^4$ ($5 \otimes 5 = 1 \oplus 3 \oplus 5 \oplus 7 \oplus 9$)

The computational cost of the tensor product $V_{\ell_1}^1 \otimes V_{\ell_2}^2$ is $O((\ell_1 + \ell_2)^3)$ due to the Clebsch-Gordan coefficients $\langle \ell_1 m_1, \ell_2 m_2 \mid \ell m \rangle$ that must be computed and applied. For $\ell_max = 2$, this is at most $5^3 = 125$ operations per feature channel per edge, which MACE reduces using a body-order truncation that avoids high-order products.

B.3 Glossary of Geometric Deep Learning Terms

The following terms appear throughout Part VII with specialised meanings that may differ from their use in other subfields of mathematics.

Equivariance. A map $f: X \to Y$ between G-spaces is G-equivariant if $f(g \cdot x) = g \cdot f(x)$ for all $g \in G$. In molecular modelling, SE(3)-equivariance means that rotating/translating the input atoms produces a correspondingly rotated/translated output.

Frame. In the context of AlphaFold 2, a frame $T = (R, t) \in SE(3)$ is a rigid coordinate system attached to each residue, defined by the positions of the backbone atoms N, Cα, C via Gram-Schmidt orthogonalisation.

Irrep. An irreducible representation of a group G is a representation with no proper invariant subspaces. For SO(3), the irreps are labelled by $\ell \in \mathbb{N}_0$ and have dimension $2\ell + 1$. Every SO(3)-equivariant network decomposes its features into a direct sum of irreps.

LDDT. Local Distance Difference Test: the per-residue confidence metric output by AlphaFold 2 (pLDDT), measuring the fraction of predicted inter-residue distances within 0.5, 1, 2, and 4 Å of the true distance, averaged over neighbours within 15 Å.

MSA. Multiple Sequence Alignment: an alignment of homologous protein sequences that encodes evolutionary co-variation. Co-varying positions tend to be in 3D contact; the Evoformer exploits this signal via axial attention and triangular updates.

Pairformer. The simplified successor to the Evoformer in AlphaFold 3, which operates on the pair representation alone (no MSA dimension) using triangle attention and multiplicative updates.

Wigner D-matrix. The matrix representation $D^{\ell}(R)$ of a rotation $R \in SO(3)$ in the ℓ-th irrep. Entries are given by $D^{\ell}_{mm'}(\alpha, \beta, \gamma)$ where α, β, γ are the Euler angles of R. Wigner D-matrices satisfy the composition rule $D^{\ell}(R_1 R_2) = D^{\ell}(R_1) D^{\ell}(R_2)$.

Appendix C: Technical Supplement for Part VIII — Agentic AI, Meta-Learning, and Continual Learning

This appendix provides extended derivations and definitions supporting Part VIII. All notation follows Appendix A.

C.1 POMDP Formalism: Extended Definitions and the Belief-State MDP

A Partially Observable Markov Decision Process (POMDP) is defined by the tuple (S, A, O, P, R, Ω, γ) introduced in Part VIII, Chapter 1. The key reduction to a belief-state MDP proceeds as follows.

Definition C.1 (Belief state). The belief state at time t is the posterior distribution $b_t \in \Delta(S)$ defined by $b_t(s) = P(S_t = s \mid o_1, a_1, \ldots, o_t)$. It is updated via Bayes' rule after each observation:

$$b'(s') = \Omega(o' \mid s', a) \, \Sigma_s P(s' \mid s, a) \, b(s) \; / \; P(o' \mid b, a)$$

This defines the belief update operator $\tau: \Delta(S) \times A \times O \to \Delta(S)$. The belief-state MDP has state space $\Delta(S)$ (a continuous simplex), action space A, and reward $\underline{R}(b, a) = \Sigma_s \, b(s) \, R(s, a)$. An optimal POMDP policy $\pi^*: \Delta(S) \to A$ maps belief states to actions and is equivalent to an optimal policy for the belief-state MDP.

Definition C.2 (Value function of a POMDP). The optimal value function $V^*: \Delta(S) \to \mathbb{R}$ satisfies the Bellman equation:

$$V^*(b) = \max_a \left[\underline{R}(b, a) + \gamma \, \Sigma_{o'} P(o' \mid b, a) \, V^*(\tau(b, a, o')) \right]$$

V^* is piecewise linear and convex in b (it is the upper envelope of a set of linear α-vectors). Exact POMDP solvers (PBVI, SARSOP) approximate V^* over a finite set of belief points. For LLM agents, the context window serves as a finite-dimensional approximation to b, with the attention mechanism implementing a soft version of the belief update operator τ.

C.2 EWC as Laplace Approximation: Complete Derivation

We derive the Elastic Weight Consolidation objective from the Bayesian perspective, filling in the steps sketched in Theorem 5.1 of Part VIII.

Starting point. After learning task $\mathcal{T}_1$, the posterior over parameters is $p(\theta \mid \mathcal{D}_1)$. When learning task $\mathcal{T}_2$, Bayes' rule gives:

$$\log p(\theta \mid \mathcal{D}_1, \mathcal{D}_2) = \log p(\mathcal{D}_2 \mid \theta) + \log p(\theta \mid \mathcal{D}_1) + \text{const}$$

Laplace approximation of $p(\theta \mid \mathcal{D}_1)$. At the MAP estimate $\theta_1^* = \arg\max p(\theta \mid \mathcal{D}_1)$, the second-order Taylor expansion gives:

$$\log p(\theta \mid \mathcal{D}_1) \approx \log p(\theta_1^* \mid \mathcal{D}_1) - (1/2)(\theta - \theta_1^*)^T H_1 (\theta - \theta_1^*)$$

where $H_1 = -\nabla^2_\theta \log p(\theta_1^* \mid \mathcal{D}_1)$ is the Hessian of the negative log-posterior at θ_1^*. Since $\log p(\theta \mid \mathcal{D}_1) = \log p(\mathcal{D}_1 \mid \theta) + \log p(\theta) - \text{const}$, and using the Fisher identity $F_1 \approx H_1$ (valid at a maximum likelihood point for a well-specified model), we obtain:

$$\log p(\theta \mid \mathcal{D}_1) \approx \text{const} - (1/2) \sum_i F_{1,ii} (\theta_i - \theta_{1,i}^*)^2$$

where the last step uses the diagonal approximation $F_1 \approx \text{diag}(F_{1,11}, \ldots, F_{1,pp})$. The MAP estimator for the joint posterior then minimises:

$$\mathcal{L}_{\text{EWC}}(\theta) = \mathcal{L}_{\{\mathcal{T}_2\}}(\theta) + (\lambda/2) \sum_i F_{1,ii} (\theta_i - \theta_{1,i}^*)^2$$

which is precisely the EWC objective. The diagonal Fisher $F_{1,ii} = \mathbb{E}_{\{p_data\}}[(\partial \log p_\theta(y \mid x) / \partial \theta_i)^2]$ at $\theta = \theta_1^*$ is computed on the task-1 training set as a sample average. ■

C.3 Topological Data Analysis: Key Definitions

This section provides precise definitions for the topological objects used in Part VIII, Chapter 6.

Definition C.3 (Simplicial complex). A simplicial complex K on a vertex set V is a collection of subsets (simplices) of V such that (i) every singleton $\{v\} \in K$, and (ii) if $\sigma \in K$ and $\tau \subseteq \sigma$, then $\tau \in K$. A k-simplex has $k + 1$ vertices. The Vietoris-Rips complex $\text{VR}_\varepsilon(X)$ at scale ε is a simplicial complex on point cloud X where a subset $\sigma \subseteq X$ is a simplex iff $\text{diam}(\sigma) \leq \varepsilon$.

Definition C.4 (Homology group). For a simplicial complex K with coefficient ring $\mathbb{Z}/2\mathbb{Z}$, the k-th homology group $H_k(K) = \ker(\partial_k) / \text{im}(\partial_{k+1})$ where ∂_k is the boundary operator mapping k-chains to $(k-1)$-chains. The rank $\beta_k = \text{rank}(H_k(K))$ is the k-th Betti number: β_0 = connected components, β_1 = independent loops, β_2 = enclosed voids.

Definition C.5 (Persistence diagram). Given a filtration $K_0 \subseteq K_1 \subseteq \ldots \subseteq K_n$, a homological feature (class $\alpha \in H_k$) is born at $\varepsilon_b = \min\{\varepsilon_i : \alpha \in \text{im}(H_k(K_i) \to H_k(K_n))\}$ and dies at $\varepsilon_d = \min\{\varepsilon_j > \varepsilon_b : \alpha \in \text{im}(H_k(K_b) \to H_k(K_j)) = 0\}$. The persistence of α is $\text{pers}(\alpha) = \varepsilon_d - \varepsilon_b$. The persistence diagram D_k is the multiset of birth-death pairs $\{(\varepsilon_b, \varepsilon_d)\}$ for all k-dimensional homological features.

Remark C.1 (Bottleneck distance and stability). The bottleneck distance $d_B(D, D') = \inf_{\gamma} \sup_{p \in D} \|p - \gamma(p)\|_\infty$ between persistence diagrams D, D' satisfies the stability theorem: if $f, g: X \to \mathbb{R}$ with $\|f - g\|_\infty \leq \varepsilon$, then $d_B(D(f), D(g)) \leq \varepsilon$. This guarantees that small perturbations of the loss landscape or network weights produce small changes in the persistence diagram, validating topological training diagnostics.

C.4 GRPO Objective: Detailed Derivation from PPO

GRPO (Group Relative Policy Optimisation) modifies PPO by replacing the learned value function baseline with a within-group reward normalisation. Here we trace the derivation carefully.

PPO baseline. PPO minimises the surrogate loss $\mathscr{L}_PPO = -\mathbb{E}[\min(r(\theta)\hat{A}, \text{clip}(r(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A})]$, where $r(\theta) = \pi_\theta(y|x) / \pi_\{\theta_old\}(y|x)$ is the probability ratio and $\hat{A} = (r - V_\varphi(x))$ is the advantage estimated using a learned value function V_φ.

GRPO substitution. For a problem x and G sampled responses $\{y_1, ..., y_G\}$ from $\pi_\{\theta_old\}$, the group-relative advantage replaces $V_\varphi(x)$ with the within-group mean reward:

$$\hat{A}_i = (r(x, y_i) - \text{mean_j } r(x, y_j)) / \text{std_j } r(x, y_j)$$

This is an unbiased estimator of the advantage: $\mathbb{E}_\{y\sim\pi\}[\hat{A}_i] = 0$ by the centring. The variance of $\hat{A}_i$ is O(1/G): larger group size G reduces variance as more within-group comparisons are made. The KL penalty $\beta\, D_KL(\pi_\theta \| \pi_ref)$ prevents the policy from drifting too far from a reference (pre-RLHF) model, playing the same role as the KL term in RLHF with PPO. The elimination of V_φ saves approximately one forward pass and one backward pass per training step (the critic network), halving memory usage in practice.

Bibliography for Part VII: Scientific AI and Geometric Deep Learning

The following references are cited in Part VII. Works are listed alphabetically by first author's surname. All arxiv preprints were accessed and verified as of publication date of this treatise.

Primary Architecture Papers

Abramson, J., Adler, J., Dunger, J., et al. (2024). Accurate structure prediction of biomolecular interactions with AlphaFold 3. Nature, 630, 493–500. doi:10.1038/s41586-024-07487-w. Introduces AlphaFold 3, extending structure prediction to DNA, RNA, and small molecules via a diffusion module operating on 3D atom coordinates.

Benegas, G., Bhodiwal, S., Ye, C., & Song, Y. S. (2024). AlphaGenome: Advancing genomic sequence and variant effect prediction. bioRxiv. Presents a unified multi-task model for regulatory genomics, predicting RNA-seq, ATAC-seq, ChIP-seq, and Hi-C from raw DNA sequence using multi-scale convolution and reverse complement equivariance.

Hayes, T., Rao, R., Akin, H., et al. (2024). Simulating 500 million years of evolution with a language model. Science, 385(6707). Introduces ESM-3, a multimodal protein language model jointly modelling sequence, structure, and function through unified masked language modelling.

Jumper, J., Evans, R., Pritzel, A., et al. (2021). Highly accurate protein structure prediction with AlphaFold. Nature, 596, 583–589. doi:10.1038/s41586-021-03819-2. The definitive paper on AlphaFold 2; introduces the Evoformer, Invariant Point Attention, FAPE loss, and the dual MSA/pair representation.

Equivariant Networks and Geometric Deep Learning

Batzner, S., Musaelian, A., Sun, L., et al. (2022). E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. Nature Communications, 13, 2453. Introduces NequIP, a highly data-efficient interatomic potential using tensor-product-based SE(3)-equivariant message passing.

Batatia, I., Kovács, D. P., Simm, G., Ortner, C., & Csányi, G. (2022). MACE: Higher order equivariant message passing neural networks for fast and accurate force fields. NeurIPS 2022. Introduces MACE, which uses body-order truncation to make high-ℓ tensor products computationally feasible.

Cohen, T. S. & Welling, M. (2016). Group equivariant convolutional networks. ICML 2016. The foundational paper establishing group-equivariant CNNs; proves that convolution is the unique linear equivariant operation for locally compact groups.

Cohen, T. S., Geiger, M., Köhler, J., & Welling, M. (2018). Spherical CNNs. ICLR 2018. Extends equivariant CNNs to the sphere S^2 using the spherical harmonics as the relevant irreducible representations of SO(3).

Klicpera, J., Groß, J., & Günnemann, S. (2020). Directional message passing for molecular graphs. ICLR 2020. Introduces DimeNet, which incorporates bond angles as directional features, strictly increasing the expressivity of distance-only GNNs.

Liao, Y. L. & Smidt, T. (2023). Equiformer: Equivariant graph attention transformer for 3D atomistic graphs. ICLR 2023. Brings the Transformer architecture to SE(3)-equivariant settings via equivariant dot-product attention.

Schütt, K., Kindermans, P.-J., Felix, H. E. S., Chmiela, S., Tkatchenko, A., & Müller, K.-R. (2017). SchNet: A continuous-filter convolutional neural network for modeling quantum interactions. NeurIPS 2017. Introduces the first scalable distance-based GNN for molecular property prediction.

Schütt, K., Unke, O., & Gastegger, M. (2021). Equivariant message passing for the prediction of tensorial properties and molecular spectra. ICML 2021. Introduces PaiNN, which augments scalar features with equivariant vector features, enabling force prediction and chiral discrimination.

Satorras, V. G., Hoogeboom, E., & Welling, M. (2021). E(n) equivariant graph neural networks. ICML 2021. A computationally lightweight alternative to spherical-harmonic-based equivariant GNNs, achieving E(n)-equivariance without explicit irrep decompositions.

Thomas, N., Smidt, T., Kearnes, S., et al. (2018). Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds. arxiv:1802.08219. The first explicit use of spherical harmonics and Clebsch-Gordan products in equivariant neural networks for molecular data.

Group Theory and Representation Theory

Bronstein, M. M., Bruna, J., Cohen, T., & Veličković, P. (2021). Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. arxiv:2104.13478. The comprehensive mathematical treatment of geometric deep learning as a unification of existing architectures through the lens of symmetry and equivariance.

Fulton, W. & Harris, J. (1991). Representation Theory: A First Course. Springer Graduate Texts in Mathematics, vol. 129. The standard reference for the representation theory of Lie groups, including SO(3), SU(2), and their Clebsch-Gordan theory.

Weyl, H. (1946). The Classical Groups: Their Invariants and Representations (2nd ed.). Princeton University Press. The classical reference connecting group invariant theory to physics and geometry, underpinning the equivariance framework.

Multi-Task Learning and Gradient Methods

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., & Finn, C. (2020). Gradient surgery for multi-task learning. NeurIPS 2020. Introduces PCGrad, the gradient projection method for resolving conflicting gradients in multi-task settings, applied to regulatory genomics in AlphaGenome.

Bibliography for Part VIII: Agentic AI, Meta-Learning, and Continual Learning

The following references are cited in Part VIII. Works are listed alphabetically by first author's surname.

Agentic Systems and Multi-Agent Reinforcement Learning

Akyürek, E., Schuurmans, D., Andreas, J., Ma, T., & Zhou, D. (2023). What learning algorithm is in-context learning? Investigations with linear models. ICLR 2023. Proves that transformer in-context learning implements ridge regression via a gradient-descent-like mechanism in the attention layers.

Chaudhry, A., Ranzato, M. A., Rohrbach, M., & Elhoseiny, M. (2019). Efficient lifelong learning with A-GEM. ICLR 2019. Introduces A-GEM, a computationally efficient approximation to GEM that projects the gradient onto a single averaged past-task constraint.

Lasry, J.-M. & Lions, P.-L. (2007). Mean field games. Japanese Journal of Mathematics, 2(1), 229–260. The foundational paper establishing mean-field game theory and the forward-backward equation system for the N→∞ limit.

Lopez-Paz, D. & Ranzato, M. A. (2017). Gradient episodic memory for continual learning. NeurIPS 2017. Introduces GEM, which enforces non-negative inner products between the current gradient and stored past-task gradients as an exact non-forgetting constraint.

Rashid, T., Samvelyan, M., Schroeder de Witt, C., Farquhar, G., Foerster, J., & Whiteson, S. (2018). QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. ICML 2018. Introduces the QMIX mixing network and the Individual-Global-Max (IGM) condition for cooperative MARL.

Yao, S., Zhao, J., Yu, D., et al. (2022). ReAct: Synergizing reasoning and acting in language models. ICLR 2023. Introduces the ReAct framework for LLM agents, interleaving chain-of-thought reasoning with tool-call actions within the context window.

Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., & Wu, Y. (2022). The surprising effectiveness of PPO in cooperative multi-agent games. NeurIPS 2022. Demonstrates that MAPPO with centralised critics is competitive with specialised MARL algorithms on StarCraft and other benchmarks.

Inference-Time Compute and Reinforcement Learning from Human Feedback

Coulom, R. (2006). Efficient selectivity and backup operators in Monte-Carlo tree search. Computers and Games, LNCS 4630. The paper introducing MCTS with the UCB1 selection rule; foundational for AlphaGo and subsequent work on MCTS for LLMs.

Lightman, H., Kosaraju, V., Burda, Y., et al. (2023). Let's verify step by step. arxiv:2305.20050. Introduces Process Reward Models trained on step-by-step human annotations,

demonstrating their superiority over outcome-only reward models for mathematical reasoning.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal Policy Optimization algorithms. arxiv:1707.06347. The PPO paper; introduces the clipped surrogate objective and the KL-penalty variant, both used in GRPO.

Shao, Z., Wang, P., Zhu, Q., et al. (2024). DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. arxiv:2402.03300. Introduces GRPO (Group Relative Policy Optimisation) and demonstrates its effectiveness for mathematical reasoning without a critic network.

Snell, C., Lee, J., Xu, K., & Kumar, A. (2024). Scaling LLM test-time compute optimally. arxiv:2408.03314. A systematic study of test-time compute scaling strategies (Best-of-N, beam search, PRM-guided MCTS), establishing the optimal allocation of inference-time compute.

Video Generation and Diffusion Models

Brooks, T., Peebles, B., Holmes, C., et al. (2024). Video generation models as world simulators. OpenAI Technical Report. Introduces Sora: a video generation model based on DiT with spatiotemporal patch tokenisation and 3D RoPE, trained to produce physically coherent videos of arbitrary resolution and duration.

Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., & Fleet, D. J. (2022). Video Diffusion Models. NeurIPS 2022. The foundational paper on diffusion models for video generation; introduces factored spatial and temporal attention to reduce the $O((THW)^2)$ computational cost of full spatiotemporal attention.

Yang, Z., Teng, J., Zheng, W., et al. (2024). CogVideoX: Text-to-video diffusion models with an expert transformer. arxiv:2408.06072. Demonstrates high-quality text-to-video synthesis using 3D full attention with flow matching in approximately 50 sampling steps.

Meta-Learning

Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. ICML 2017. Introduces MAML; proves that a single gradient step on a small support set, followed by a meta-gradient through this step, produces fast task adaptation across diverse problem domains.

Nichol, A., Achiam, J., & Schulman, J. (2018). On first-order meta-learning algorithms. arxiv:1803.02999. Introduces Reptile and FOMAML; proves that Reptile implicitly maximises within-task gradient alignment, providing a first-order justification for MAML-like meta-learning.

Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. NeurIPS 2017. Introduces ProtoNets, where class prototypes are the means of support-set embeddings and classification uses nearest-prototype in Euclidean space.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., & Wierstra, D. (2016). Matching networks for one shot learning. NeurIPS 2016. The seminal few-shot learning paper

introducing attention-based matching over a support set, predating and motivating subsequent Transformer-based approaches.

Continual Learning

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., et al. (2017). Overcoming catastrophic forgetting in neural networks. Proceedings of the National Academy of Sciences, 114(13), 3521–3526. Introduces Elastic Weight Consolidation (EWC), using the Fisher information matrix diagonal as parameter importance weights in an L2-regularisation term that penalises forgetting.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., et al. (2016). Progressive neural networks. arxiv:1606.04671. Introduces Progressive Neural Networks, which add a new column of parameters per task with lateral connections to previous frozen columns, guaranteeing zero forgetting by construction.

Zenke, F., Poole, B., & Ganguli, S. (2017). Continual learning through synaptic intelligence. ICML 2017. Introduces Synaptic Intelligence (SI), an online variant of EWC that accumulates parameter importance weights during training based on each parameter's contribution to the loss decrease.

Topological Analysis of Learning

Carlsson, G. (2009). Topology and data. Bulletin of the American Mathematical Society, 46(2), 255–308. The foundational survey of persistent homology and topological data analysis; introduces the persistence diagram and the stability theorem.

Cohen-Steiner, D., Edelsbrunner, H., & Harer, J. (2007). Stability of persistence diagrams. Discrete and Computational Geometry, 37(1), 103–120. Proves the bottleneck stability theorem (Theorem 6.1 of Part VIII), establishing that persistence diagrams vary continuously with the underlying function.

Frankle, J. & Carlin, M. (2019). The lottery ticket hypothesis: Finding sparse, trainable neural networks. ICLR 2019. Empirically identifies sparse subnetworks ("winning tickets") that match the performance of dense networks when trained in isolation, motivating the study of loss landscape geometry.

Nanda, N., Chan, L., Lieberum, T., Smith, J., & Steinhardt, J. (2023). Progress measures for grokking via mechanistic interpretability. ICLR 2023. Provides the mechanistic account of grokking for modular arithmetic; identifies the Fourier-circular representation and the phase transition in H_1 persistent homology as the mechanism of delayed generalisation.

Naitzat, G., Zhitnikov, A., & Lim, L.-H. (2020). Topology of deep neural networks. Journal of Machine Learning Research, 21(184), 1–40. Analyses the topological transformation of data representations across layers, showing that deep networks progressively reduce the Betti numbers of class boundaries.

MATHEMATICS OF DEEP LEARNING

SUPPLEMENTARY VOLUME

APPENDICES & BIBLIOGRAPHY

Mathematical Notation · Key Theorems · Proof Techniques

Architecture Taxonomy · Glossary · Full References I–VIII

Eight-Part Monograph Series on the Mathematical Principles of Neural Networks

Independent of Companion Volumes — Complete in Itself

Silvano Salvador

© Mathematics of Deep Learning Series

SUPPLEMENTARY VOLUME

APPENDICES & BIBLIOGRAPHY

Appendix A: Mathematical Notation and Conventions
Appendix B: Compendium of Key Theorems
Appendix C: Core Proof Techniques
Appendix D: Taxonomy of Neural Architectures
Appendix E: Glossary of Technical Terms
Bibliography: Parts I–VIII

© Mathematics of Deep Learning Series

Silvano Salvador

APPENDICES

Mathematical Notation · Theorems Reference · Proof Techniques · Index

Based on the work of Silvano Salvador

Appendix A: Mathematical Notation and Conventions

This appendix consolidates the notational conventions used throughout the eight parts of the treatise. The notation follows that of Salvador's Sezione 0, augmented as needed by standard conventions from the literature. Ambiguity is resolved in favour of clarity over brevity.

A.1 Sets, Spaces, and Structural Objects

Symbol	Meaning / Definition
$\mathbb{R}$, $\mathbb{C}$, $\mathbb{Z}$, $\mathbb{N}$	Real numbers, complex numbers, integers, natural numbers (0-inclusive)
$\mathbb{R}^n$, $\mathbb{R}^{m\times n}$	n-dimensional real vector space; space of real m×n matrices
$\mathbb{S}^{n-1}$	Unit (n-1)-sphere in $\mathbb{R}^n$: $\{x \in \mathbb{R}^n : \|x\|_2 = 1\}$
$\Delta(X)$	Set of probability distributions over X (probability simplex if X finite)
$\mathcal{H}$	Hilbert space; $\mathcal{G}$ denotes a general measurable space
$\mathcal{X}$, $\mathcal{Y}$	Input space, output space (generic)
$[n]$	The set {1, 2, ..., n}
$\mathcal{P}(X)$	Power set of X; $\mathcal{P}_k(X)$ = all k-element subsets
G, H	Groups (script letters); $g \in G$ denotes a group element
$GL(n)$, $O(n)$, $SO(n)$	General linear, orthogonal, special orthogonal groups over $\mathbb{R}$
$SE(3)$	Special Euclidean group in 3D: rigid motions (rotations + translations)
$\mathcal{G} = (V, E)$	Graph with vertex set V and edge set E

A.2 Vectors and Matrices

Symbol	Meaning
x, y, z	Column vectors (lower-case bold italic in equations)
x_i	i-th component of vector x; x_{ij} is (i,j) entry of matrix X
X, W, A, B	Matrices (upper-case); W^{(ℓ)} is the weight matrix at layer ℓ
X^{S}	Transpose of matrix X; X† (or X*) = conjugate transpose (Hermitian)
X^{-1}, X†	Matrix inverse; Moore-Penrose pseudoinverse
⊗	Kronecker product: (A ⊗ B)_{ij,kl} = A_{ik} B_{jl}
⊙	Hadamard (element-wise) product: (A ⊙ B)_{ij} = A_{ij} B_{ij}
diag(v)	Diagonal matrix with entries given by vector v
vec(X)	Column-major vectorisation of matrix X
Tr(X)	Trace of square matrix X = $\Sigma_i X_{ii}$
rank(X)	Rank (dimension of column space) of X
σ_i(X)	i-th singular value of X (σ_1 ≥ σ_2 ≥ ... ≥ 0)
‖X‖_F	Frobenius norm = (Σ_{ij} X^2_{ij})^{1/2} = (Σ_i σ_i^2)^{1/2}
‖X‖_*	Nuclear norm = Σ_i σ_i (sum of singular values)
$\|X\|_2$	Spectral norm = σ_1(X) (largest singular value)
‖x‖_p	ℓ^p-norm = (Σ_i $\|x_i\|^p$)^{1/p}; ‖x‖_∞ = max_i $\|x_i\|$
⟨x, y⟩	Inner product = $x^T y$ (Euclidean); ⟨f, g⟩_$\mathcal{H}$ for Hilbert space inner product

A.3 Calculus and Optimisation

Symbol	Meaning
∂f/∂x	Partial derivative of scalar f with respect to x
∇_θ f(θ)	Gradient vector of f at θ: $(\partial f/\partial\theta_1, ..., \partial f/\partial\theta_p)^T$
∇^2_θ f(θ)	Hessian matrix of f: H_{ij} = $\partial^2 f / \partial\theta_i \partial\theta_j$
J_f(x)	Jacobian of vector-valued map f: (J_f)_{ij} = ∂f_i/∂x_j

Symbol	Meaning
δL/δf	Functional derivative of functional L with respect to function f
∂_θ L	Shorthand for ∇_θ L (log-derivative notation)
d/dt, $\dot{\mathbf{x}}$	Time derivative of x(t)
∘	Function composition: (f ∘ g)(x) = f(g(x))
f*	Convex conjugate (Legendre-Fenchel transform): f*(y) = sup_x {⟨x,y⟩ - f(x)}
prox_{λf}(x)	Proximal operator: arg min_z {f(z) + (1/2λ)‖z - x‖²}
∂f(x)	Subdifferential of convex f at x (set of subgradients)
O(·), Θ(·), Ω(·)	Landau asymptotic notation (upper, tight, lower bounds)

A.4 Probability and Statistics

Symbol	Meaning
p(x), q(x)	Probability density or mass function (context determines which)
P(A)	Probability of event A
𝔼_p[f(X)]	Expectation of f(X) under distribution p; 𝔼[·] when p is clear from context
Var_p[X]	Variance = 𝔼[(X - 𝔼X)²]
Cov(X,Y)	Covariance = 𝔼[(X - 𝔼X)(Y - 𝔼Y)^T]
$\mathcal{N}$(μ, Σ)	Multivariate Gaussian with mean μ ∈ $\mathbb{R}^n$ and covariance Σ ∈ ℝ^{n×n}
$\mathcal{N}$(0, I_d)	Standard d-dimensional Gaussian
Cat(π)	Categorical distribution with probability vector π ∈ Δ^K
Ber(p)	Bernoulli distribution with success probability p
σ(x)	Logistic sigmoid: σ(x) = 1/(1 + e^{-x})
softmax(z)_i	Softmax: exp(z_i) / Σ_j exp(z_j)

Symbol	Meaning
lse(β, z)	Log-sum-exp: (1/β) log Σᵢ exp(β z_i)
D_KL(p ‖ q)	KL divergence: 𝔼_p[log p(X)/q(X)] ≥ 0; equals 0 iff p = q a.e.
H(p)	Entropy: -𝔼_p[log p(X)]; H(p,q) = -𝔼_p[log q(X)] = cross-entropy
I(X;Y)	Mutual information: H(X) - H(X∣Y) = D_KL(p(x,y) ‖ p(x)p(y))
W_p(μ, ν)	p-Wasserstein distance between probability measures μ, ν
D_TV(p, q)	Total variation distance: (1/2) Σ_x ∣p(x) - q(x)∣
x ~ p	'x is distributed according to p' / 'x is sampled from p'
x ⊥ y ∣ z	Conditional independence of x and y given z

A.5 Deep Learning–Specific Notation

Symbol	Meaning
L(θ), ℒ(θ)	Loss function (L) and ELBO / variational objective ($\mathscr{L}$)
f_θ	Neural network with parameters θ; f_θ: $\mathscr{X} \to \mathscr{Y}$
a^{(ℓ)}, z^{(ℓ)}	Post-activation and pre-activation at layer ℓ
δ^{(ℓ)}	Error signal at layer ℓ during backpropagation
W^{(ℓ)}, b^{(ℓ)}	Weight matrix and bias vector at layer ℓ
σ(·)	Activation function (context: logistic sigmoid or generic nonlinearity)
ReLU(x)	Rectified linear unit: max(0, x); component-wise for vectors
GELU(x)	Gaussian error linear unit: x · Φ(x) where Φ is Gaussian CDF
Q, K, V	Query, Key, Value matrices in attention mechanism
Attn(Q,K,V)	Scaled dot-product attention: softmax($QK^T/\sqrt{d}$) · V
G(θ)	Fisher Information Matrix (FIM): 𝔼[∇log p · ∇log p^T]
Θ(x,x')	Neural Tangent Kernel (NTK) evaluated at inputs x, x'
π_θ(a∣s)	Policy: probability of action a in state s with parameters θ

Symbol	Meaning
$V^\pi(s)$, $Q^\pi(s,a)$	Value function and action-value function under policy π
$\mathcal{T}^\pi$, $\mathcal{T}^*$	Bellman evaluation and optimality operators
r_t, G_t	Reward at time t; return (discounted sum of rewards from t)
$\mathcal{D}$	Dataset or replay buffer
ε_t	Reparametrisation noise $\varepsilon \sim \mathcal{N}(0,I)$ in VAE / diffusion
$s_\theta(x,t)$	Score network approximating $\nabla_x \log p_t(x)$

Appendix B: Compendium of Key Theorems

This appendix provides a self-contained statement of the principal theorems referenced across the eight parts of the treatise, organised by mathematical domain. For each theorem, the originating chapter and key references are given. Proofs are sketched where the technique is instructive; complete proofs are found in the cited sources.

B.1 Linear Algebra and Matrix Theory

Theorem B.1 (Singular Value Decomposition)

Every matrix $A \in \mathbb{R}^{m \times n}$ admits the factorisation $A = U\Sigma V^T$ where $U \in O(m)$, $V \in O(n)$, and $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_r, 0, \ldots, 0)$ with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$, $r = \mathrm{rank}(A)$.

The SVD is unique (up to sign) when the singular values are distinct.

Best rank-k approximation (Eckart-Young): $\|A - \hat{A}_k\|_F$ is minimised by $\hat{A}_k = U_k \Sigma_k V_k^T$ (first k columns of U, k×k top-left of Σ, first k columns of V). The error is $\|A - \hat{A}_k\|_F = (\Sigma_{i>k} \sigma_i^2)^{1/2}$.

Theorem B.2 (Marchenko-Pastur Law)

Let $X \in \mathbb{R}^{n \times p}$ have i.i.d. entries with mean 0 and variance σ^2/n; let $\gamma = p/n \to c \in (0, \infty)$.

The empirical spectral distribution of $X^T X$ converges weakly a.s. to the Marchenko-Pastur law:

$$dF_c(\lambda) = \sqrt{(b-\lambda)(\lambda-a)} \,/\, (2\pi c \sigma^2 \lambda) \cdot 1_{[a,b]}(\lambda) \;+\; \max(0, 1-1/c) \cdot \delta_0$$

where $a = \sigma^2(1 - \sqrt{c})^2$, $b = \sigma^2(1 + \sqrt{c})^2$.

Deviations from MP (heavy-tailed bulk, outlier eigenvalues) signal learned structure in trained networks.

Theorem B.3 (Weyl's Perturbation Theorem)

Let $A, E \in \mathbb{R}^{n \times n}$ be symmetric. Denote eigenvalues $\lambda_1(\cdot) \geq \ldots \geq \lambda_n(\cdot)$. Then:

$$\max_i \, |\lambda_i(A + E) - \lambda_i(A)| \leq \|E\|_2$$

Application: The Hessian $\nabla^2 L(\theta)$ of the loss is perturbed by mini-batch noise; the sharpness (largest Hessian eigenvalue) can change by at most the spectral norm of the noise. This bounds the rate of sharpness increase during training — a tool for analysing flat minimum methods (SAM).

B.2 Information Theory and Information Geometry

Theorem B.4 (Data Processing Inequality)

For any Markov chain $X \to Y \to Z$ (Y is a function of X, Z is a function of Y):

$I(X; Z) \le I(X; Y)$ and $I(X; Z) \le I(Y; Z)$

Mutual information cannot increase through processing. Consequence: no representation Z of input X can contain more information about label Y than X itself — the theoretical ceiling for learning is $I(X;Y)$.

Theorem B.5 (Fisher-Cramér-Rao Bound)

For any unbiased estimator $T(X_1,...,X_n)$ of a parameter $\theta \in \mathbb{R}^d$ from i.i.d. observations:

$\mathrm{Cov}(T) \ge G(\theta)^{-1} / n$ (in the Loewner / positive semidefinite sense)

where $G(\theta)$ is the Fisher Information Matrix. The bound is achieved by the MLE in regular families. Role in ML: the FIM is the metric on the statistical manifold; natural gradient descent is coordinate-invariant steepest descent in the FIM metric, not the Euclidean metric.

Theorem B.6 (Gibbs' Variational Principle)

For any distribution q and any measurable f:

$\log \mathbb{E}_p[\exp(f)] = \sup_q \{ \mathbb{E}_q[f] - D_{KL}(q \| p) \}$

The supremum is attained at $q^*(x) \propto p(x) \exp(f(x))$.

Consequences: (1) The ELBO identity in VAEs; (2) the optimal RLHF policy ($\pi^* \propto \pi_{ref} \cdot \exp(r/\beta)$); (3) the variational representation of the log-partition function in energy-based models.

B.3 Statistical Learning Theory

Theorem B.7 (PAC Learning Generalisation Bound via Rademacher Complexity)

Let $\mathscr{H}$ be a hypothesis class, $\mathscr{S} = \{(x_i, y_i)\}_{i=1}^m$ i.i.d. from D. With probability $\ge 1-\delta$:

for all $h \in \mathscr{H}$: $L_D(h) \le \hat{L}_S(h) + 2\mathfrak{R}_m(\mathscr{H}) + \sqrt{(\log(2/\delta) / 2m)}$

where $\mathbb{E}_m(\in) = {}_i _\sigma[\sup_{\{h_i\}}] (1/m) \Sigma_i \sigma_i h(x_i)$] is the empirical Rademacher complexity.

For neural networks with bounded weight norms, $\mathfrak{R}_m \le C \cdot \|W\|_F / \sqrt{m}$ (matrix spectral norms).

Theorem B.8 (Neural Tangent Kernel Convergence)

Consider a fully-connected network $f_\theta: \mathbb{R}^{n_0} \to \mathbb{R}$ with widths $n_1 = \ldots = n_L = n \to \infty$.
The NTK $\Theta_n(x,x') = \Sigma_p (\partial f_\theta(x)/\partial\theta_p)(\partial f_\theta(x')/\partial\theta_p)$ converges in probability to a deterministic kernel $\Theta_\infty(x,x')$ as $n \to \infty$. Under gradient flow on the squared loss, if the NTK Gram matrix $K = (\Theta_\infty(x_i,x_j))$
has minimum eigenvalue $\lambda_{min} > 0$, the training loss converges exponentially:

$$L(t) \le L(0) \cdot \exp(-2\lambda_{min} t)$$

B.4 Optimisation

Theorem B.9 (Convergence of SGD for Non-convex Functions)

Let $f: \mathbb{R}^d \to \mathbb{R}$ be L-smooth ($\|\nabla f(x) - \nabla f(y)\| \le L\|x-y\|$) and bounded below.
SGD with step size $\eta_t = \eta/\sqrt{t}$, where gradient estimates satisfy $\mathbb{E}[\|\hat{g}_t - \nabla f(\theta_t)\|^2] \le \sigma^2$:

$$(1/T)\ \Sigma_{t=1}^{T}\ \mathbb{E}[\|\nabla f(\theta_t)\|^2] \le O(\sigma / \sqrt{T}) + O(L\ \eta\ \sigma^2)$$

So after $T = O(\varepsilon^{-4})$ steps, expected gradient norm is $\le \varepsilon$ (first-order stationarity).
Adam achieves the same rate with per-coordinate learning rates, often with better constants.

Theorem B.10 (Mirror Descent and Natural Gradient as Bregman Projections)

Natural gradient descent is equivalent to Mirror Descent with the Bregman divergence
$D_F(\theta, \theta') = F(\theta) - F(\theta') - \langle \nabla F(\theta'), \theta - \theta' \rangle$ where F is the log-partition function:

$$\theta_{t+1} = \arg\min_\theta \{\ \eta\ \langle \nabla L(\theta_t), \theta \rangle + D_{KL}(p_\theta\ \|\ p_{\theta_t})\ \}$$

This makes explicit that natural gradient descent follows the steepest descent path in the Riemannian manifold of distributions, not in parameter space.

B.5 Generative Models

Theorem B.11 (ELBO Decomposition for VAEs)

For any distribution $q_\varphi(z|x)$ (approximate posterior) and model $p_\theta(x,z) = p(z)p_\theta(x|z)$:

$$\log p_\theta(x) = \mathcal{L}(\theta,\varphi;x) + D_{KL}(q_\varphi(z|x)\ \|\ p_\theta(z|x))$$

where $\mathscr{L} = \mathbb{E}_{q_\varphi}[\log p_\theta(x|z)] - D_{KL}(q_\varphi(z|x) \| p(z))$ is the ELBO.
Since $D_{KL} \geq 0$, the ELBO is a lower bound: $\log p_\theta(x) \geq \mathscr{L}(\theta,\varphi;x)$.
Maximising $\mathscr{L}$ jointly over θ, φ simultaneously maximises the log-likelihood lower bound and minimises the approximation gap between q_φ and the true posterior.

Theorem B.12 (Anderson Reverse-Time SDE)

Let $\{x_t\}_{t\in[0,T]}$ satisfy the forward SDE: $dx_t = f(x_t,t)dt + g(t)dW_t$
with marginal densities $p_t(x)$. The exact reverse-time SDE (Anderson 1982) is:

$$dx_t = [f(x_t,t) - g(t)^2 \nabla_x \log p_t(x_t)]\, dt + g(t)\, d\bar{W}_t$$

where $\bar{W}_t$ is a reverse-time Brownian motion.
The score function $\nabla_x \log p_t$ can be approximated by a neural network $s_\theta(x,t)$ trained via denoising score matching, converting learning the reverse SDE to supervised regression.

Theorem B.13 (Wasserstein GAN Duality)

By the Kantorovich-Rubinstein theorem, the 1-Wasserstein distance between p_{data} and p_θ is:

$$W_1(p_{data}, p_\theta) = \sup_{f:\, \mathrm{Lip}(f) \leq 1} \{ \mathbb{E}_{x\sim p_{data}}[f(x)] - \mathbb{E}_{x\sim p_\theta}[f(x)] \}$$

Wasserstein GANs optimise this by parameterising the 1-Lipschitz critic f with a neural network (weight clipping or gradient penalty to enforce the Lipschitz constraint), and training the generator to minimise W_1 rather than JS divergence. The advantage: W_1 is continuous even when supports are disjoint.

B.6 Reinforcement Learning

Theorem B.14 (Bellman Contraction and Value Iteration Convergence)

The Bellman optimality operator $\mathbb{R}^*: \to^S \mathbb{R} \circledcirc ^S$ is a γ-contraction in $\|\cdot\|_\infty$:

$$\|\mathscr{T}^*V - \mathscr{T}^*U\|_\infty \leq \gamma \, \|V - U\|_\infty \quad \text{for all } V, U \in \mathbb{R}^S$$

By Banach's fixed-point theorem, $V^* = \lim_{k\to\infty} (\mathscr{T}^*)^k V_0$ for any V_0, with

$$\|(\mathscr{T}^*)^k V_0 - V^*\|_\infty \leq \gamma^k \, \|V_0 - V^*\|_\infty$$

Policy Iteration converges in finitely many steps (at most $|A|^{|S|}$ policies).

Theorem B.15 (DPO Equivalence to RLHF, Rafailov et al. 2023)

Under the Bradley-Terry preference model $p(y^+ \succ y^- | x) = \sigma(r^*(x,y^+) - r^*(x,y^-))$,
the policy that minimises $L_{DPO}(\theta) = -\mathbb{E}[\log \sigma(\beta \log \pi_\theta(y^+|x)/\pi_{ref}(y^+|x) - \beta \log \pi_\theta(y^-|x)/\pi_{ref}(y^-|x))]$
is identical to the optimal policy of the KL-constrained RL problem:
$\pi^* = \arg\max_\pi \mathbb{E}[r^*(x,y)] - \beta D_{KL}(\pi \| \pi_{ref})$
DPO thus implicitly fits a reward model and optimises the RL objective simultaneously, without requiring an explicit reward model or RL training loop.

B.7 Algebraic Topology and Statistical Physics

Theorem B.16 (Persistent Homology Stability)

Let $f, g: \mathcal{K} \to \mathbb{R}$ be functions on a simplicial complex $\mathcal{K}$. The bottleneck distance between their persistence diagrams satisfies:
$d_B(PD(f), PD(g)) \leq \|f - g\|_\infty$
Consequence: persistent homology is a stable topological summary. Small perturbations in the loss function cannot produce large changes in the topology of its sublevel filtration. This makes persistent homology a robust tool for comparing loss landscapes across training runs.

Theorem B.17 (Hopfield Network Capacity: Classical vs Modern)

Classical Hopfield (binary): $M_{max} = 0.138\, N$ random patterns retrievable
Modern Hopfield (continuous): $M_{max} = \exp(\alpha N)$ patterns for some $\alpha > 0$
The exponential increase stems from replacing the bilinear energy with a log-sum-exp energy, which is equivalent to computing attention scores. The update rule becomes $\sigma_{new} = \Xi\, \text{softmax}(\beta\, \Xi^T \sigma)$, identifying Modern Hopfield retrieval with Transformer attention.

Appendix C: Core Proof Techniques

This appendix catalogues the principal proof techniques that recur throughout the treatise, providing brief explanations and examples of their use. Mastery of these techniques is prerequisite to reading the research literature.

C.1 The Log-Derivative (Score Function) Trick

Whenever an expectation $\mathbb{E}_{p_\theta}[f(x)]$ must be differentiated with respect to θ, the log-derivative identity $\nabla_\theta p_\theta(x) = p_\theta(x) \nabla_\theta \log p_\theta(x)$ converts the derivative of the integral into an integral:

$$\nabla_\theta \mathbb{E}_{p_\theta}[f(x)] = \mathbb{E}_{p_\theta}[f(x) \nabla_\theta \log p_\theta(x)]$$

This enables Monte Carlo gradient estimation by sampling $x \sim p_\theta$. Uses: REINFORCE, variational inference BBVI, policy gradient theorem. The resulting estimator is unbiased but high-variance; control variates (baselines) reduce variance without introducing bias.

C.2 The Reparametrisation Trick

For continuous distributions where $x = g_\theta(\varepsilon)$ with $\varepsilon \sim p(\varepsilon)$ independent of θ (e.g., $x = \mu_\theta + \sigma_\theta \varepsilon$, $\varepsilon \sim \mathcal{N}(0,I)$), the gradient passes through the sampling operation:

$$\nabla_\theta \mathbb{E}_{p_\theta(x)}[f(x)] = \mathbb{E}_{p(\varepsilon)}[\nabla_\theta f(g_\theta(\varepsilon))]$$

The reparametrisation trick decouples the stochastic node from the parameter θ, enabling low-variance gradients. Uses: VAE, SAC, normalising flows, continuous diffusion. Fails for discrete distributions — the Gumbel-Softmax relaxation provides a biased but low-variance alternative.

C.3 McDiarmid's Inequality and Uniform Convergence

McDiarmid's inequality bounds the deviation of a function of independent random variables:

$$\text{If } f(X_1,\ldots,X_m) \text{ satisfies: } \sup_{x_i,x'_i} |f(\ldots,x_i,\ldots) - f(\ldots,x'_i,\ldots)| \le c_i$$

$$\text{then: } P(f(X) - \mathbb{E}[f(X)] \ge t) \le \exp(-2t^2 / \textstyle\sum_i c_i^2)$$

Uses: all PAC learning generalisation bounds. The function f is the empirical risk; the bounded difference condition holds with $c_i = 2/m$ for bounded losses. Combined with a union bound over $\mathcal{H}$ (or chaining for infinite $\mathcal{H}$), this yields uniform convergence bounds.

C.4 The Replica Method (Statistical Physics Heuristic)

The replica method computes quenched averages $\mathbb{E}_J[\log Z(J)]$ via the identity $\log Z = \lim_{n\to 0} \partial/\partial n\, \mathbb{E}[Z^n]$. For integer n, $Z^n = \Pi_{a=1}^{n} Z$ creates n independent copies ('replicas') that become coupled through the disorder average. After computing $\mathbb{E}[Z^n]$ analytically for integer n and identifying order parameters Q_{ab}, one analytically continues to $n \to 0$.

The method is non-rigorous (the $n \to 0$ limit lacks a general justification) but produces exact results that have subsequently been rigorously verified. Uses: Hopfield capacity, SK spin glass free energy, learning curve of linear classifiers, high-dimensional regression.

C.5 Stochastic Approximation and the ODE Method

Many RL and ML algorithms are stochastic approximation iterations: $\theta_{t+1} = \theta_t + \alpha_t\, h(\theta_t, \xi_t)$ where ξ_t is noise and h is the update direction. Convergence is established by showing that the 'mean ODE' $d\theta/dt = H(\theta) := \mathbb{E}[h(\theta, \xi)]$ has a globally attracting fixed point, and the stochastic iterates shadow this ODE (Ljung 1977, Borkar & Meyn 2000).

Key conditions: (1) Robbins-Monro step sizes $\Sigma\, \alpha_t = \infty$, $\Sigma\, \alpha_t^2 < \infty$; (2) h is bounded on compact sets; (3) the mean ODE is Lyapunov stable. Uses: Q-learning convergence (Watkins-Dayan), TD(λ) convergence, stochastic gradient methods.

Appendix D: Taxonomy of Neural Architectures and Inductive Biases

The following table organises major architectural families by their symmetry group (inductive bias), representative models, primary domain, and references to relevant chapters of the treatise.

Architecture Family	Symmetry / Inductive Bias	Representative Models	Domain	Treatise Part
MLP	None (dense)	FFN, MLP-Mixer	General	I
CNN	Translation equivariance	ResNet, EfficientNet, ConvNeXt	Vision	I, III
Transformer	Permutation equivariance + positional	GPT, BERT, ViT, Swin	Language, Vision	I, II
GNN	Graph isomorphism	GCN, GAT, MPNN, GraphTransformer	Graphs	II
LSTM/RNN	Sequential order	LSTM, GRU, Mamba, S4	Sequences	II
Equivariant Net	SE(3)/E(3) equivariance	SE(3)-TFN, EGNN, NequIP	Molecular/3D	I, II
VAE	Disentangled latent structure	β-VAE, NVAE, HiVAE	Generative	IV
GAN	Adversarial training	StyleGAN, BigGAN, CycleGAN	Generative	IV
NF	Invertibility + tractable Jacobian	RealNVP, Glow, NSF	Generative	IV
Diffusion	Reverse SDE	DDPM, LDM, DiT, SiT	Generative	IV
EBM	Energy function	EBM, JEM, Modern Hopfield	Generative/AM	IV, VI
RL Policy	Markov, temporal structure	PPO, SAC, TD3, AlphaZero	Control	V

Appendix E: Glossary of Technical Terms

A concise glossary of technical terms used throughout the treatise, with brief formal definitions.

Attention. A weighted sum of values, where weights are computed as a softmax of dot products between queries and keys. The core operation of Transformer models.

Backpropagation. Efficient computation of gradients in a computational graph via the chain rule, traversed in reverse topological order (backward mode autodiff).

Bayesian Inference. Updating a prior distribution $p(\theta)$ to a posterior $p(\theta \mid \text{data}) \propto p(\text{data} \mid \theta)p(\theta)$ using Bayes' theorem; used in BNNs and Laplace approximation.

Contrastive Loss. A loss that pushes together representations of matched pairs and pushes apart unmatched pairs; InfoNCE is the mutual-information-maximising variant.

Diffusion Model. A generative model that learns to reverse a Gaussian noising process via score matching or denoising score matching.

ELBO. Evidence Lower BOund: a tractable lower bound on $\log p(x)$ used in VAE and variational inference; ELBO = $\mathbb{E}_q[\log p(x \mid z)] - D_{KL}(q \| p)$.

Emergent Capability. A capability absent in small models and present in large models, appearing discontinuously as scale increases.

Equivariance. A map f is equivariant to group G if $f(gx) = gf(x)$ for all $g \in G$. Invariance is the special case $f(gx) = f(x)$.

Fisher Information Matrix. $G(\theta) = \mathbb{E}[\nabla \log p \cdot \nabla \log p^{T}]$; the Riemannian metric on the statistical manifold; curvature of the KL divergence to second order.

Flow Matching. A generative model that learns a vector field $v_\theta(x,t)$ whose ODE $dx/dt = v_\theta$ transports a source distribution to the data distribution.

Grokking. The phenomenon where a network memorises training data but later (after extended training) discovers a generalising algorithm.

Hilbert Space. A complete inner product space; RKHS is the Hilbert space of functions associated to a kernel, central to kernel methods and SVMs.

Importance Sampling. Estimating $\mathbb{E}_p[f]$ by sampling from q: $\mathbb{E}_p[f] = \mathbb{E}_q[f \cdot p/q]$. Used in off-policy RL (importance ratio π/π_{old}).

Inductive Bias. A constraint built into model architecture that makes certain hypotheses more likely, typically reflecting domain knowledge.

KL Divergence. $D_{KL}(p \| q) = \mathbb{E}_p[\log(p/q)] \geq 0$; measures 'extra bits' needed to encode p using code optimised for q; asymmetric.

LoRA. Low-Rank Adaptation: fine-tuning by learning $\Delta W = BA$ with $r \ll \min(m,n)$; reduces trainable parameters from mn to $r(m+n)$.

MCMC. Markov Chain Monte Carlo: algorithms (MH, HMC, NUTS) for sampling from unnormalised distributions by constructing ergodic Markov chains.

Mutual Information. $I(X;Y) = H(X) - H(X \mid Y) = D_{KL}(p(x,y) \| p(x)p(y))$; measures statistical dependence; central to information bottleneck and contrastive learning.

Natural Gradient. $\nabla^{nat}_\theta L = G(\theta)^{-1} \nabla_\theta L$; steepest descent in the Fisher-Rao metric; invariant under reparametrisation of θ.

NTK. Neural Tangent Kernel: the limiting kernel of an infinite-width network, constant during gradient flow training (lazy training regime).

Optimal Transport. $W_p(\mu,\nu)$: the cost of the cheapest way to transport mass from μ to ν; the p-Wasserstein distance; used in GANs, flow matching, and alignment.

Persistent Homology. Topological summary of a filtration of simplicial complexes; records birth and death of connected components, loops, and voids as scale varies.

Rademacher Complexity. $\mathfrak{R}_m(\mathscr{H})$: a data-dependent measure of hypothesis class capacity that tightly bounds generalisation; based on correlation with random sign patterns.

Replica Method. A statistical physics heuristic for computing $\mathbb{E}[\log Z]$ via analytic continuation from integer n (replicas) to $n \to 0$.

Residual Connection. $x_{\ell+1} = x_\ell + F(x_\ell)$; additive skip connection that stabilises gradient flow by providing a gradient highway of slope 1.

Score Function. $\nabla_x \log p(x)$; the gradient of the log-density; learned by score-matching and used to drive the reverse diffusion SDE.

Superposition. The hypothesis that neural networks represent more features than neurons by storing them as near-orthogonal directions in activation space.

VC Dimension. The largest set of points that can be shattered (all 2^n labellings achieved) by some hypothesis; measures expressiveness of a hypothesis class.

Wasserstein Distance. $W_p(\mu,\nu) = (\inf_{\gamma \in \Gamma(\mu,\nu)} \int c^p \, d\gamma)^{1/p}$; a proper metric on probability distributions that metrises weak convergence.

BIBLIOGRAPHY

Based on the work of Silvano Salvador

Bibliographic Note

This bibliography provides the complete references for claims, theorems, and empirical results cited across all parts of the treatise. Entries are organised thematically, mirroring the structure of the treatise itself, with a final section for the foundational source volumes by Silvano Salvador. Within each section, entries are ordered roughly chronologically to convey the intellectual history of each subfield.

Citation tags use the first three letters of the first author's surname followed by a two-digit year (e.g., [Vap95] for Vapnik 1995). In case of collision, a letter suffix disambiguates (e.g., [Sch17a] vs [Sch17b]). Preprints on arXiv are cited by their arXiv identifier where journal publication was not available at the time of writing.

I. Foundations: Linear Algebra, Optimisation, and Automatic Differentiation

Linear Algebra and Matrix Theory

[GVL13] G.H. Golub, C.F. Van Loan. *Matrix Computations, 4th ed.*. Johns Hopkins University Press, 2013. Standard reference for numerical linear algebra, SVD, and eigendecompositions.

[Bha97] R. Bhatia. *Matrix Analysis*. Springer Graduate Texts in Mathematics, 1997. Comprehensive treatment of spectral theory, perturbation bounds, and matrix functions.

[Wey12] H. Weyl. *Das asymptotische Verteilungsgesetz der Eigenwerte linearer partieller Differentialgleichungen*. Mathematische Annalen, 71(4), 1912. Classical perturbation bound for eigenvalues; modern statement in Bhatia (1997).

[Mar67] V.A. Marchenko, L.A. Pastur. *Distribution of eigenvalues for some sets of random matrices*. Mathematics of the USSR-Sbornik, 1(4), 1967. Foundational RMT result for empirical spectral distribution of sample covariance matrices.

[Mar18] C.H. Martin, M.W. Mahoney. *Implicit Self-Regularization in Deep Neural Networks*. arXiv:1810.01075, 2018. Heavy-tail self-regularisation hypothesis; power-law spectral analysis of trained networks.

[Voi85] D. Voiculescu. *Symmetries of some reduced free product C*-algebras*. Springer LNM 1132, 1985. Introduction to free probability theory; S-transform multiplicativity.

[And10] G.W. Anderson, A. Guionnet, O. Zeitouni. *An Introduction to Random Matrices*. Cambridge University Press, 2010. Standard graduate textbook on RMT, Wigner semicircle, MP law, free probability.

Backpropagation and Automatic Differentiation

[Rum86] D.E. Rumelhart, G.E. Hinton, R.J. Williams. *Learning representations by back-propagating errors*. Nature, 323, 1986. Classic statement of backpropagation for multilayer networks.

[Bay88] Y. Le Cun. *A theoretical framework for back-propagation*. Proceedings of the 1988 Connectionist Models Summer School, 1988. Automatic differentiation perspective on backpropagation.

[Bay15] A.G. Baydin, B.A. Pearlmutter, A.A. Radul, J.M. Siskind. *Automatic Differentiation in Machine Learning: a Survey*. JMLR, 18(153), 2018. Comprehensive survey of forward and reverse mode AD, and their ML applications.

[Pas19] A. Paszke et al.. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. NeurIPS 2019, 2019. PyTorch autodiff engine; dynamic computation graph and eager execution.

Optimisation

[Kin15] D.P. Kingma, J. Ba. *Adam: A Method for Stochastic Optimization.* ICLR 2015, 2015. Adam optimizer; adaptive moment estimation; bias correction.

[Los17] I. Loshchilov, F. Hutter. *Decoupled Weight Decay Regularization.* ICLR 2019, 2017. AdamW; shows that standard L2 and Adam weight decay are not equivalent.

[Bot10] L. Bottou. *Large-Scale Machine Learning with Stochastic Gradient Descent.* COMPSTAT 2010, 2010. Theoretical foundations of SGD for large-scale ML; learning rate schedules.

[Nes83] Y. Nesterov. *A method for solving the convex programming problem with convergence rate O(1/k²).* Soviet Mathematics Doklady, 27(2), 1983. Nesterov momentum; accelerated gradient descent for convex objectives.

[Mar20] D. Masters, C. Luschi. *Revisiting Small Batch Training for Deep Neural Networks.* arXiv:1804.07612, 2018. Empirical study of batch size effects on training dynamics and generalisation.

[For18] P. Foret, A. Kleiner, H. Mobahi, B. Neyshabur. *Sharpness-Aware Minimization for Efficiently Improving Generalization.* ICLR 2021, 2021. SAM: seeking flat minima by minimising the worst-case loss in a parameter neighbourhood.

[Ama98] S. Amari. *Natural Gradient Works Efficiently in Learning.* Neural Computation, 10(2), 1998. Natural gradient descent; Fisher information metric; invariance under reparametrisation.

[Mar15] J. Martens, R. Grosse. *Optimizing Neural Networks with Kronecker-factored Approximate Curvature.* ICML 2015, 2015. K-FAC: Kronecker-factored approximation of the Fisher information matrix.

II. Architectures and Geometric Inductive Biases

Transformer Architecture

[Vas17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin. *Attention Is All You Need.* NeurIPS 2017, 2017. Original Transformer architecture; multi-head self-attention; encoder-decoder.

[Dev19] J. Devlin, M.W. Chang, K. Lee, K. Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* NAACL 2019, 2019. Masked language modelling pre-training; bidirectional attention.

[Bro20] T.B. Brown et al.. *Language Models are Few-Shot Learners.* NeurIPS 2020, 2020. GPT-3 (175B); few-shot and zero-shot in-context learning; scaling to 175B parameters.

[Dao22] T. Dao, D. Fu, S. Ermon, A. Rudra, C. Ré. *FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness.* NeurIPS 2022, 2022. IO-aware tiled attention; reduces memory from $O(S^2)$ to $O(S)$; enables long contexts.

[Dao23] T. Dao. *FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning.* ICLR 2024, 2023. Improved work distribution and parallelism for modern GPU architectures.

[Su24] J. Su, Y. Lu, S. Pan, A. Murtadha, B. Wen, Y. Liu. *RoFormer: Enhanced Transformer with Rotary Position Embedding.* Neurocomputing, 2024. RoPE: rotary position embeddings; multiplicative position encoding compatible with linear attention.

[Kat23] A. Katharopoulos, A. Vyas, N. Pappas, F. Fleuret. *Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention.* ICML 2020, 2020. Linear attention via kernel feature maps; reduces attention complexity from $O(S^2)$ to $O(S)$.

[Mit23] D. De, S. Smith, A. Fernando et al.. *Griffin: Mixing Gated Linear Recurrences with Local Attention.* arXiv:2402.19427, 2024. Hybrid linear-recurrence plus local attention; efficient long-sequence modelling.

Equivariant and Geometric Architectures

[Coh16] T.S. Cohen, M. Welling. *Group Equivariant Convolutional Networks.* ICML 2016, 2016. G-CNNs; equivariance to discrete symmetry groups; group convolution theorem.

[Tho18] N. Thomas et al.. *Tensor Field Networks: Rotation- and Translation-Equivariant Neural Networks for 3D Point Clouds.* arXiv:1802.08219, 2018. SE(3)-equivariant networks via spherical harmonic tensor products.

[Jum21] J. Jumper et al.. *Highly accurate protein structure prediction with AlphaFold.* Nature, 596, 2021. AlphaFold 2; IPA mechanism; FAPE loss; SE(3)-equivariance in protein structure prediction.

[Bro22] J.L. Batzner et al.. *E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials.* Nature Communications, 13, 2022. NequIP; SE(3)-equivariant GNNs for molecular dynamics force fields.

Graph Neural Networks

[Kip17] T.N. Kipf, M. Welling. *Semi-Supervised Classification with Graph Convolutional Networks.* ICLR 2017, 2017. GCN; spectral graph convolution; Chebyshev polynomial approximation.

[Vel18] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio. *Graph Attention Networks.* ICLR 2018, 2018. GAT; attention-weighted aggregation over graph neighbourhoods.

[Xu19] K. Xu, W. Hu, J. Leskovec, S. Jegelka. *How Powerful are Graph Neural Networks?.* ICLR 2019, 2019. WL-test equivalence of MPNNs; GIN architecture achieving maximum WL-expressiveness.

[Bod22] C. Bodnar, F. Di Giovanni, B. Chamberlain, P. Liò, M. Bronstein. *Neural Sheaf Diffusion: A Topological Perspective on Heterophily in GNNs.* NeurIPS 2022, 2022. Cellular sheaves on graphs; sheaf Laplacian; sheaf neural networks.

Sequential and State Space Models

[Hoc97] S. Hochreiter, J. Schmidhuber. *Long Short-Term Memory.* Neural Computation, 9(8), 1997. LSTM; gated cell state; gradient highways through forget gate.

[Gu22] A. Gu, K. Goel, C. Ré. *Efficiently Modeling Long Sequences with Structured State Spaces.* ICLR 2022, 2022. S4; HiPPO matrices; FFT-based SSM convolution; O(L log L) computation.

[Gu23] A. Gu, T. Dao. *Mamba: Linear-Time Sequence Modeling with Selective State Spaces.* arXiv:2312.00752, 2023. Selective SSM; content-dependent B,C matrices; hardware-aware parallel scan.

[Pen23] B. Peng et al.. *RWKV: Reinventing RNNs for the Transformer Era.* EMNLP 2023, 2023. RWKV; linear attention as recurrence; WKV operator.

[Pol23] M. Poli et al.. *Hyena Hierarchy: Towards Larger Convolutional Language Models.* ICML 2023, 2023. Hyena; subquadratic long convolutions; implicit parameterisation.

III. Statistical Learning Theory and Probabilistic Methods

Learning Theory

[Vap95] V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995. VC dimension, SRM, SVM; PAC learning framework.

[Bar02] P.L. Bartlett, S. Mendelson. *Rademacher and Gaussian Complexities: Risk Bounds and Structural Results*. JMLR, 3, 2002. Rademacher complexity; data-dependent generalisation bounds; tighter than VC.

[Ney15] B. Neyshabur, R. Tomioka, N. Srebro. *Norm-Based Capacity Control in Neural Networks*. COLT 2015, 2015. Spectral norm bounds for neural networks; parameter norm vs capacity.

[Bel20] M. Belkin, D. Hsu, S. Ma, S. Mandal. *Reconciling modern machine-learning practice and the classical bias-variance tradeoff*. PNAS, 116(32), 2019. Double descent; benign overfitting; interpolation threshold.

[Jac18] A. Jacot, F. Gabriel, C. Hongler. *Neural Tangent Kernel: Convergence and Generalization in Neural Networks*. NeurIPS 2018, 2018. NTK; infinite-width convergence; lazy training regime.

[Aro19] S. Arora, S. Du, W. Hu, Z. Li, R. Wang. *Fine-Grained Analysis of Optimization and Generalization for Overparameterized Two-Layer Neural Networks*. ICML 2019, 2019. NTK convergence guarantees for two-layer networks; overparameterised regime.

Bayesian Methods and MCMC

[Nea93] R.M. Neal. *Probabilistic Inference Using Markov Chain Monte Carlo Methods*. Technical Report CRG-TR-93-1, University of Toronto, 1993. Foundational reference for HMC in ML; MCMC for neural networks.

[Hof14] M.D. Hoffman, A. Gelman. *The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo*. JMLR, 15(1), 2014. NUTS; adaptive HMC; No-U-Turn criterion; basis for Stan's sampler.

[Wel11] M. Welling, Y.W. Teh. *Bayesian Learning via Stochastic Gradient Langevin Dynamics*. ICML 2011, 2011. SGLD; stochastic gradient MCMC; injecting gradient noise for posterior sampling.

[Mac03] D.J.C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003. Comprehensive Bayesian ML reference; Laplace approximation; free energy.

Variational Inference and Uncertainty Quantification

[Kin14] D.P. Kingma, M. Welling. *Auto-Encoding Variational Bayes*. ICLR 2014, 2014. VAE; ELBO; reparametrisation trick; amortised variational inference.

[Res15] D.J. Rezende, S. Mohamed. *Variational Inference with Normalizing Flows*. ICML 2015, 2015. Normalising flows for flexible approximate posteriors; planar/radial flows.

[Bur16] Y. Burda, R. Grosse, R. Salakhutdinov. *Importance Weighted Autoencoders*. ICLR 2016, 2016. IWAE; tighter ELBO via importance-weighted samples; gap to log p(x).

[Ang19] A. Angelopoulos, S. Bates. *A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification*. arXiv:2107.07511, 2021. Conformal prediction; distribution-free coverage guarantees for any predictor.

[Lac17] B. Lakshminarayanan, A. Pritzel, C. Blundell. *Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles*. NeurIPS 2017, 2017. Deep ensembles; calibrated uncertainty via ensemble disagreement.

IV. Generative Models

GANs

[Goo14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio. *Generative Adversarial Networks.* NeurIPS 2014, 2014. Original GAN; minimax game; JS divergence minimisation.

[Arj17] M. Arjovsky, S. Chintala, L. Bottou. *Wasserstein Generative Adversarial Networks.* ICML 2017, 2017. WGAN; Kantorovich-Rubinstein duality; gradient penalty for Lipschitz constraint.

[Kar20] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, T. Aila. *Analyzing and Improving the Image Quality of StyleGAN.* CVPR 2020, 2020. StyleGAN2; style-based generator; path length regularisation; FID records.

Normalising Flows

[Die14] L. Dinh, D. Krueger, Y. Bengio. *NICE: Non-linear Independent Components Estimation.* ICLR Workshop 2015, 2014. NICE; additive coupling layers; first tractable normalising flow.

[Die17] L. Dinh, J. Sohl-Dickstein, S. Bengio. *Density estimation using Real-valued Non-Volume Preserving (Real NVP) transformations.* ICLR 2017, 2017. RealNVP; affine coupling layers; bijective generative model.

[Kin18] D.P. Kingma, P. Dhariwal. *Glow: Generative Flow with Invertible 1×1 Convolutions.* NeurIPS 2018, 2018. Glow; learned permutation via invertible 1×1 convolutions; high-quality image synthesis.

[Dur19] C. Durkan, A. Bekasov, I. Murray, G. Papamakarios. *Neural Spline Flows.* NeurIPS 2019, 2019. NSF; monotone rational-quadratic splines as coupling layers; expressive normalising flow.

Score-Based Models and Diffusion

[Son19] Y. Song, S. Ermon. *Generative Modeling by Estimating Gradients of the Data Distribution.* NeurIPS 2019, 2019. Score matching with Langevin dynamics; annealed IS for sample quality.

[Ho20] J. Ho, A. Jain, P. Abbeel. *Denoising Diffusion Probabilistic Models.* NeurIPS 2020, 2020. DDPM; Gaussian diffusion; denoising score matching as variational lower bound.

[Son21] Y. Song, J. Sohl-Dickstein, D.P. Kingma, A. Kumar, S. Ermon, B. Poole. *Score-Based Generative Modeling through Stochastic Differential Equations.* ICLR 2021, 2021. Unification of SMLD and DDPM via SDEs; VE/VP/subVP SDEs; probability flow ODE.

[Son22] J. Song, C. Meng, S. Ermon. *Denoising Diffusion Implicit Models*. ICLR 2021, 2021. DDIM; deterministic sampling from diffusion models; 10–50× inference speedup.

[Lip22] W. Lipman, R. Chen, H. Ben-Hamu, M. Nickel, M. Le. *Flow Matching for Generative Modeling*. ICLR 2023, 2022. Flow Matching; conditional vector fields; OT-FM for straight flows.

[Rom22] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, B. Ommer. *High-Resolution Image Synthesis with Latent Diffusion Models*. CVPR 2022, 2022. LDM / Stable Diffusion; diffusion in VQ-regularised latent space; cross-attention conditioning.

[Son23] Y. Song, P. Dhariwal, M. Chen, I. Sutskever. *Consistency Models*. ICML 2023, 2023. Consistency models; self-consistency property; direct one-step generation.

Self-Supervised and Contrastive Learning

[Che20] T. Chen, S. Kornblith, M. Norouzi, G. Hinton. *A Simple Framework for Contrastive Learning of Visual Representations*. ICML 2020, 2020. SimCLR; NT-Xent loss; data augmentation for contrastive self-supervised learning.

[Gri20] J.B. Grill et al.. *Bootstrap Your Own Latent: A New Approach to Self-Supervised Learning*. NeurIPS 2020, 2020. BYOL; self-distillation without negative pairs; online/target network asymmetry.

[Bar21] J. Zbontar, L. Jing, I. Misra, Y. LeCun, S. Deny. *Barlow Twins: Self-Supervised Learning via Redundancy Reduction*. ICML 2021, 2021. Barlow Twins; cross-correlation matrix towards identity; information redundancy.

Energy-Based Models and Hopfield

[Ram21] H. Ramsauer, B. Schäfl, J. Lehner, P. Seidl, M. Widrich, L. Gruber, M. Holzleitner, M. Pavlović, G.K. Sandve, V. Greiff, D. Kreil, M. Kopp, G. Klambauer, J. Brandstetter, S. Hochreiter. *Hopfield Networks is All You Need*. ICLR 2021, 2021. Modern Hopfield networks; exponential capacity; equivalence with attention.

[Lec06] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, F. Huang. *A Tutorial on Energy-Based Learning*. Predicting Structured Data, MIT Press, 2006. EBMs; energy function learning; contrastive divergence.

V. Reinforcement Learning and Alignment

MDP Theory and Value-Based Methods

[Put94] M.L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, 1994. Standard MDP reference; existence and uniqueness of optimal policy; PI convergence.

[Wat92] C.J.C.H. Watkins, P. Dayan. *Q-learning*. Machine Learning, 8(3-4), 1992. Q-learning algorithm; convergence proof via stochastic approximation theory.

[Mni15] V. Mnih et al.. *Human-level control through deep reinforcement learning*. Nature, 518, 2015. DQN; experience replay; target network; Atari benchmark.

[Bel17] M.G. Bellemare, W. Dabney, R. Munos. *A Distributional Perspective on Reinforcement Learning*. ICML 2017, 2017. Distributional RL; C51; Wasserstein contraction; return distribution.

Policy Gradient Methods

[Wil92] R.J. Williams. *Simple statistical gradient-following algorithms for connectionist reinforcement learning*. Machine Learning, 8(3-4), 1992. REINFORCE; score function gradient estimator; baseline variance reduction.

[Sut00] R.S. Sutton, D.A. McAllester, S.P. Singh, Y. Mansour. *Policy Gradient Methods for Reinforcement Learning with Function Approximation*. NeurIPS 2000, 2000. Policy Gradient Theorem; actor-critic; compatible function approximation.

[Sch16] J. Schulman, P. Moritz, S. Levine, M. Jordan, P. Abbeel. *High-Dimensional Continuous Control Using Generalized Advantage Estimation*. ICLR 2016, 2016. GAE; bias-variance tradeoff in advantage estimation; λ interpolation.

[Sch15] J. Schulman, S. Levine, P. Abbeel, M. Jordan, P. Moritz. *Trust Region Policy Optimization*. ICML 2015, 2015. TRPO; KL-constrained policy update; Fisher-vector products via conjugate gradient.

[Sch17] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov. *Proximal Policy Optimization Algorithms*. arXiv:1707.06347, 2017. PPO; clipped importance ratio surrogate; simpler and more robust than TRPO.

MaxEnt RL and Continuous Control

[Haa18] T. Haarnoja, A. Zhou, P. Abbeel, S. Levine. *Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor*. ICML 2018, 2018. SAC; soft Bellman equations; entropy-regularised RL; automatic temperature tuning.

[Fuj18] S. Fujimoto, H. van Hoof, D. Meger. *Addressing Function Approximation Error in Actor-Critic Methods.* ICML 2018, 2018. TD3; clipped double Q; target policy smoothing; delayed policy updates.

[Lil16] T.P. Lillicrap et al.. *Continuous control with deep reinforcement learning.* ICLR 2016, 2016. DDPG; deterministic policy gradient; actor-critic for continuous action spaces.

Model-Based RL

[Jan19] M. Janner, J. Fu, M. Zhang, S. Levine. *When to Trust Your Model: Model-Based Policy Optimization.* NeurIPS 2019, 2019. MBPO; theoretical bound on model error; short model rollouts for safe data augmentation.

[Haf19] D. Hafner, T. Lillicrap, J. Ba, M. Norouzi. *Dream to Control: Learning Behaviors by Latent Imagination.* ICLR 2020, 2019. Dreamer; RSSM; latent imagination; policy optimisation via imagined trajectories.

[Haf23] D. Hafner, J. Lillicrap, P. Norouzi, T.P. Lillicrap. *Mastering Diverse Domains through World Models.* arXiv:2301.04104, 2023. DreamerV3; universal hyperparameters; symlog predictions; free-bits KL.

[Sil18] D. Silver et al.. *A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play.* Science, 362, 2018. AlphaZero; MCTS + neural network; self-play without human knowledge.

[Sch20] J. Schrittwieser et al.. *Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model.* Nature, 588, 2020. MuZero; latent dynamics model; model-based planning without observation reconstruction.

RLHF, Alignment, and Preference Learning

[Chr17] P.F. Christiano, J. Leike, T.B. Brown, M. Martic, S. Legg, D. Amodei. *Deep Reinforcement Learning from Human Preferences.* NeurIPS 2017, 2017. RLHF; preference learning from human comparisons; reward model training.

[Sti20] N. Stiennon et al.. *Learning to summarize from human feedback.* NeurIPS 2020, 2020. RLHF for summarisation; PPO + reward model; first large-scale NLP application.

[Ouy22] L. Ouyang et al.. *Training language models to follow instructions with human feedback.* NeurIPS 2022, 2022. InstructGPT; RLHF pipeline for instruction following; PPO-based fine-tuning.

[Raf23] R. Rafailov, A. Sharma, E. Mitchell, C.D. Manning, S. Ermon, C. Finn. *Direct Preference Optimization: Your Language Model is Secretly a Reward Model.* NeurIPS 2023, 2023. DPO; reward reparametrisation; elimination of explicit RL stage from RLHF.

[Eth24] K. Ethayarajh, W. Xu, N. Muennighoff, J. Jurafsky, D. Kiela. *KTO: Model Alignment as Prospect Theoretic Optimization.* ICML 2024, 2024. KTO; Kahneman-Tversky prospect theory loss; binary feedback without preference pairs.

[Men24] Y. Meng, M. Xia, D. Chen. *SimPO: Simple Preference Optimization with a Reference-Free Reward.* NeurIPS 2024, 2024. SimPO; length-normalised reward; margin-based loss; reference-free alignment.

[Bai22] Y. Bai et al.. *Constitutional AI: Harmlessness from AI Feedback.* arXiv:2212.08073, 2022. Constitutional AI; RLAIF; critique-revision self-improvement; scalable oversight.

[Gao22] L. Gao, J. Schulman, J. Hilton. *Scaling Laws for Reward Model Overoptimization.* arXiv:2210.10760, 2022. KL-return Pareto frontier; Goodhart's law in RLHF; reward hacking dynamics.

Emergent Capabilities and In-Context Learning

[Wei22] J. Wei et al.. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models.* NeurIPS 2022, 2022. CoT prompting; intermediate reasoning steps; arithmetic, commonsense, symbolic tasks.

[Wei23] J. Wei, Y. Tay, R. Bommasani et al.. *Emergent Abilities of Large Language Models.* TMLR 2022, 2022. Operational definition of emergence; phase transitions in capability with scale.

[Xie22] S.M. Xie, A. Raghunathan, P. Liang, T. Ma. *An Explanation of In-context Learning as Implicit Bayesian Inference.* ICLR 2022, 2022. ICL as Bayesian inference over latent concepts; posterior concentration with examples.

[Aky22] E. Akyürek, D. Schuurmans, J. Andreas, T. Ma, D. Zhou. *What learning algorithm is in-context learning? Investigations with linear models.* ICLR 2023, 2022. ICL as gradient descent; Transformer layers implement one step of ridge regression.

[Fen23] N. Feng, Q. Zhou, C. Gu, H. He. *Towards Revealing the Mystery behind Chain of Thought: a Theoretical Perspective.* NeurIPS 2023, 2023. CoT and circuit complexity; TC^0 vs NC^1; depth-multiplying effect of chain of thought.

[Lew20] P. Lewis et al.. *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks.* NeurIPS 2020, 2020. RAG; dense retrieval + generation; open-domain QA.

Mechanistic Interpretability

[Ola20] C. Olah, N. Cammarata, L. Schubert, G. Goh, M. Petrov, S. Carter. *Zoom In: An Introduction to Circuits.* Distill, 2020. Circuits framework; curve detectors; multimodal neurons; circuit analysis in CNNs.

[Ols22] K. Olsson, N. Elhage, N. Nanda, N. Joseph, N. DasSarma et al.. *In-context Learning and Induction Heads.* Transformer Circuits Thread, 2022. Induction heads; two-layer circuit; phase transition during training.

[Elh22] N. Elhage, T. Henighan, A. Joseph, N. Nanda, N. Olsson et al.. *Toy Models of Superposition*. Transformer Circuits Thread, 2022. Superposition hypothesis; polytope geometry of feature representation; phase transitions.

[Cun23] H. Cunningham, A. Ewart, L. Riggs, R. Huben, L. Sharkey. *Sparse Autoencoders Find Highly Interpretable Features in Language Models*. arXiv:2309.08600, 2023. SAEs for mechanistic interpretability; monosemantic feature discovery.

VI. Advanced Mathematical Structures

Statistical Physics of Learning

[Hop82] J.J. Hopfield. *Neural networks and physical systems with emergent collective computational abilities.* PNAS, 79(8), 1982. Classical Hopfield network; energy function; memory capacity and retrieval.

[Ama87] D.J. Amit, H. Gutfreund, H. Sompolinsky. *Statistical Mechanics of Neural Networks near Saturation.* Annals of Physics, 173(1), 1987. Replica calculation of Hopfield capacity; AGS critical load $\alpha_c \approx 0.138$.

[Mez87] M. Mézard, G. Parisi, M. Virasoro. *Spin Glass Theory and Beyond.* World Scientific, 1987. Replica method; Parisi RSB; SK model exact solution; spin glass theory.

[Eng01] A. Engel, C. Van den Broeck. *Statistical Mechanics of Learning.* Cambridge University Press, 2001. Statistical physics approach to ML; storage capacity; generalisation in perceptrons.

[Bor22] B. Bordelon, C. Pehlevan. *Spectral Bias Outside the Neural Tangent Kernel Regime.* ICLR 2022, 2022. Spectral learning curves; power-law scaling from data covariance spectrum.

Algebraic Topology for ML

[Ede08] H. Edelsbrunner, J. Harer. *Persistent Homology — a Survey.* Contemp. Mathematics 453, 2008. Persistent homology; filtrations; persistence diagrams; Betti numbers.

[Coh07] D. Cohen-Steiner, H. Edelsbrunner, J. Harer. *Stability of Persistence Diagrams.* Discrete & Computational Geometry, 37(1), 2007. Bottleneck stability theorem; robustness of persistent homology to perturbations.

[Car09] G. Carlsson. *Topology and Data.* Bulletin of the AMS, 46(2), 2009. TDA survey; mapper algorithm; applications to real data.

[Han20] J. Hansen, R. Ghrist. *Opinion Dynamics on Discourse Sheaves.* SIAM J. Applied Mathematics, 81(5), 2020. Cellular sheaves on graphs; sheaf Laplacian; connection Laplacian.

[Pow22] A. Power, Y. Gur-Ari, D. Bahri, R. Anderson, J. Steinhardt. *Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets.* arXiv:2201.02177, 2022. Grokking phenomenon; delayed generalisation; algorithmic datasets.

Causal Inference and OOD Generalisation

[Pea09] J. Pearl. *Causality: Models, Reasoning, and Inference, 2nd ed.*. Cambridge University Press, 2009. SCM; do-calculus; interventional and counterfactual distributions.

[Pet17] J. Peters, D. Janzing, B. Schölkopf. *Elements of Causal Inference*. MIT Press, 2017. Causal models; identifiability; ICM principle; invariant causal prediction.

[Arj19] M. Arjovsky, L. Bottou, I. Gulrajani, D. Lopez-Paz. *Invariant Risk Minimization*. arXiv:1907.02893, 2019. IRM; invariant predictors across environments; causal feature selection.

[Sag20] S. Sagawa, P.W. Koh, T.B. Hashimoto, P. Liang. *Distributionally Robust Neural Networks for Group Shifts*. ICLR 2020, 2020. Group DRO; worst-group accuracy; underrepresented group robustness.

Scaling Laws and Emergence

[Kap20] J. Kaplan, S. McCandlish, T. Henighan et al.. *Scaling Laws for Neural Language Models*. arXiv:2001.08361, 2020. Power-law scaling in N, D, C; compute-optimal allocation; chinchilla precursor.

[Hof22] J. Hoffmann et al.. *Training Compute-Optimal Large Language Models*. NeurIPS 2022, 2022. Chinchilla scaling law; equal scaling of N and D; 20 tokens per parameter.

[Sch23] R. Schaeffer, B. Miranda, S. Koyejo. *Are Emergent Abilities of Large Language Models a Mirage?*. NeurIPS 2023, 2023. Emergence as metric artefact; smooth improvement under linear metrics.

Large-Scale Fine-Tuning and Distributed Training

[Hu21] E.J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen. *LoRA: Low-Rank Adaptation of Large Language Models*. ICLR 2022, 2021. LoRA; low-rank weight perturbation; efficient fine-tuning of LLMs.

[Det23] T. Dettmers, A. Pagnoni, A. Holtzman, L. Zettlemoyer. *QLoRA: Efficient Finetuning of Quantized LLMs*. NeurIPS 2023, 2023. QLoRA; 4-bit NF4 quantisation; double quantisation; 65B on single GPU.

[Raj20] S. Rajbhandari, J. Rasley, O. Ruwase, Y. He. *ZeRO: Memory Optimizations Toward Training Trillion Parameter Models*. SC 2020, 2020. ZeRO stages 1/2/3; parameter, gradient, and optimizer state sharding.

[Zha23] Y. Zhao, A. Gu, R. Varma, L. Luo et al.. *PyTorch FSDP: Experiences on Scaling Fully Sharded Data Parallel*. VLDB 2023, 2023. FSDP; ZeRO-3 in PyTorch; per-unit parameter sharding; forward/backward gather.

[Che16] T. Chen, B. Xu, C. Zhang, C. Guestrin. *Training Deep Nets with Sublinear Memory Cost*. arXiv:1604.06174, 2016. Gradient checkpointing; √L memory-compute tradeoff; activation recomputation.

Multimodal Models

[Rad21] A. Radford, J.W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, I. Sutskever. *Learning Transferable Visual Models from Natural Language Supervision*. ICML 2021, 2021. CLIP; contrastive image-text pre-training; zero-shot transfer; InfoNCE loss.

[Ala22] J.B. Alayrac et al.. *Flamingo: a Visual Language Model for Few-Shot Learning*. NeurIPS 2022, 2022. Flamingo; Perceiver Resampler; cross-attention VLM; few-shot multimodal learning.

[Liu23] H. Liu, C. Li, Q. Wu, Y.J. Lee. *Visual Instruction Tuning*. NeurIPS 2023, 2023. LLaVA; linear projector from CLIP to LLM; multimodal instruction tuning.

[Cut13] M. Cuturi. *Sinkhorn Distances: Lightspeed Computation of Optimal Transport Distances*. NeurIPS 2013, 2013. Entropic regularised OT; Sinkhorn algorithm; efficient W distance computation.

VII. Source Volumes: The Salvador Cycle

This treatise is a synthesis and commentary on the following eleven volumes by Silvano Salvador, which constitute the primary source material. Each volume is identified by its title and the sections it covers from the Salvador numbering system.

Volume	Title	Coverage	Topics
Matematica delle reti neurali (Volume 1)	Linear Algebra for Neural Networks	Sections §1–§90	Vectors, matrices, eigendecompositions, SVD, dimensionality reduction.
Matematica delle reti neurali (Volume 2)	Calculus, Backpropagation, and Optimisation	Sections §91–§200	Gradient descent, backpropagation, Adam, loss landscapes, normalisation.
Matematica delle reti neurali (Volume 3)	Statistical Learning Theory	Sections §201–§280	VC dimension, Rademacher complexity, PAC learning, NTK, double descent.
Matematica delle reti neurali (Volume 4)	Information Geometry	Sections §281–§360	Fisher metric, natural gradient, K-FAC, information bottleneck, RKHS.
Matematica delle reti neurali (Volume 5)	Architectures and Equivariance	Sections §361–§450	Transformer, attention variants, CNN equivariance, GNN, WL-test.
Matematica delle reti neurali (Volume 6)	Generative Models I	Sections §451–§530	VAE, GAN, normalising flows, score matching, DDPM, DDIM.
Matematica delle reti neurali (Volume 7)	Generative Models II and Diffusion	Sections §531–§600	SDE diffusion, consistency models, flow matching, LDM, EBM, Hopfield.

Volume	Title	Coverage	Topics
Matematica delle reti neurali (Volume 8)	Reinforcement Learning	Sections §601–§660	MDP, Bellman, policy gradients, SAC, TD3, MBRL, RLHF, DPO.
Matematica delle reti neurali (Volume 9)	Bayesian and Probabilistic Methods	Sections §661–§720	MCMC, HMC, NUTS, SGLD, VI, ELBO, BBVI, IWAE, BNN, conformal prediction.
Matematica delle reti neurali (Volume 10)	Advanced Mathematics	Sections §721–§790	Spin glasses, replicas, free probability, algebraic topology, causal inference.
Matematica delle reti neurali (Volume 11)	Scaling, Fine-Tuning, and Multimodal	Sections §791–§860	Scaling laws, emergence, LoRA, QLoRA, ZeRO, FSDP, CLIP, VLMs.

The section numbering §1–§860 is the unified cross-volume numbering used throughout the present treatise to pinpoint specific results within the Salvador cycle.

Appendix B: Technical Supplement for Part VII — Scientific AI and Geometric Deep Learning

This appendix collects detailed proofs, derivations, and reference tables that support the main text of Part VII. All notation follows Appendix A. Equation numbers of the form (B.x) refer exclusively to this appendix.

B.1 Detailed Proof of IPA SE(3) Invariance (Theorem 2.2 of Part VII)

We provide a self-contained proof that the Invariant Point Attention (IPA) attention weights are invariant under any global rigid transformation $g = (R, t) \in SE(3)$, following Jumper et al. (2021) with minor notational adjustments.

Setup. Let $T_i = (R_i, t_i) \in SE(3)$ be the rigid frame assigned to residue i, so that the global rotation R and translation t act by $T_i \mapsto g \circ T_i = (RR_i, Rt_i + t)$. For each attention head h, residue i has query points $\tilde{q}_{ih} \in \mathbb{R}^3$ expressed in the local frame of residue i, and key points $\tilde{k}_{jh} \in \mathbb{R}^3$ expressed in the local frame of residue j. Their global-frame positions are:

$$p_{ih} = R_i \tilde{q}_{ih} + t_i, \quad k_{jh} = R_j \tilde{k}_{jh} + t_j.$$

Step 1: Distance under g. After the global transformation $g = (R, t)$, the new global positions are $p'_{ih} = R(R_i \tilde{q}_{ih}) + (Rt_i + t)$ and similarly for k'_{jh}. The squared distance between the transformed query and key points is:

$$\| p'_{ih} - k'_{jh} \|^2 = \| R(R_i \tilde{q}_{ih} - R_j \tilde{k}_{jh}) \|^2 = \| R_i \tilde{q}_{ih} - R_j \tilde{k}_{jh} \|^2 = \| p_{ih} - k_{jh} \|^2$$

where the second equality uses the fact that $R \in SO(3)$ is an isometry ($\|Rv\| = \|v\|$ for all v). The translational term cancels because both p' and k' receive the same translation $Rt_i + t$ and $Rt_j + t$ respectively, whose difference is $R(t_i - t_j)$, which is absorbed into the rotation-invariant norm.

Step 2: Invariance of attention logits. The IPA attention logit between residues i and j is:

$$a_{ij} = (1/\sqrt{(d_h + \gamma m)}) \, [q_i^{T} k_j + (\gamma m/2) \, \Sigma_h \, \| p_{ih} - k_{jh} \|^2]$$

The first term $q_i^{T} k_j$ is a dot product between sequence-space features and is independent of the 3D frames entirely. The second term is a sum of squared distances in global frame, which Step 1 shows is invariant under g. Therefore a_{ij} is SE(3)-invariant, and so are the softmax weights $A_{ij} = \mathrm{softmax}_j(a_{ij})$. ■

B.2 Clebsch-Gordan Decomposition for $\ell \leq 2$

The following table gives the irreducible decomposition of the tensor product $V_{l_1}^{1} \otimes V_{l_2}^{2}$ for angular momentum values $\ell_1, \ell_2 \le 2$, relevant to the equivariant message-passing architectures NequIP, MACE, and SEGNN. The triangle rule $|\ell_1 - \ell_2| \le \ell \le \ell_1 + \ell_2$ determines the output types.

$\ell_1 = 0, \ell_2 = 0$: $V^0 \otimes V^0 = V^0$ (scalar $\otimes$ scalar = scalar)

$\ell_1 = 0, \ell_2 = 1$: $V^0 \otimes V^1 = V^1$ (scalar $\otimes$ vector = vector)

$\ell_1 = 1, \ell_2 = 1$: $V^1 \otimes V^1 = V^0 \oplus V^1 \oplus V^2$ ($3 \otimes 3 = 1 \oplus 3 \oplus 5$)

$\ell_1 = 1, \ell_2 = 2$: $V^1 \otimes V^2 = V^1 \oplus V^2 \oplus V^3$ ($3 \otimes 5 = 3 \oplus 5 \oplus 7$)

$\ell_1 = 2, \ell_2 = 2$: $V^2 \otimes V^2 = V^0 \oplus V^1 \oplus V^2 \oplus V^3 \oplus V^4$ ($5 \otimes 5 = 1 \oplus 3 \oplus 5 \oplus 7 \oplus 9$)

The computational cost of the tensor product $V_{l_1}^{1} \otimes V_{l_2}^{2}$ is $O((\ell_1 + \ell_2)^3)$ due to the Clebsch-Gordan coefficients $\langle \ell_1 m_1, \ell_2 m_2 \mid \ell m \rangle$ that must be computed and applied. For $\ell_max = 2$, this is at most $5^3 = 125$ operations per feature channel per edge, which MACE reduces using a body-order truncation that avoids high-order products.

B.3 Glossary of Geometric Deep Learning Terms

The following terms appear throughout Part VII with specialised meanings that may differ from their use in other subfields of mathematics.

Equivariance. A map $f: X \to Y$ between G-spaces is G-equivariant if $f(g \cdot x) = g \cdot f(x)$ for all $g \in G$. In molecular modelling, SE(3)-equivariance means that rotating/translating the input atoms produces a correspondingly rotated/translated output.

Frame. In the context of AlphaFold 2, a frame $T = (R, t) \in SE(3)$ is a rigid coordinate system attached to each residue, defined by the positions of the backbone atoms N, Cα, C via Gram-Schmidt orthogonalisation.

Irrep. An irreducible representation of a group G is a representation with no proper invariant subspaces. For SO(3), the irreps are labelled by $\ell \in \mathbb{N}_0$ and have dimension $2\ell + 1$. Every SO(3)-equivariant network decomposes its features into a direct sum of irreps.

LDDT. Local Distance Difference Test: the per-residue confidence metric output by AlphaFold 2 (pLDDT), measuring the fraction of predicted inter-residue distances within 0.5, 1, 2, and 4 Å of the true distance, averaged over neighbours within 15 Å.

MSA. Multiple Sequence Alignment: an alignment of homologous protein sequences that encodes evolutionary co-variation. Co-varying positions tend to be in 3D contact; the Evoformer exploits this signal via axial attention and triangular updates.

Pairformer. The simplified successor to the Evoformer in AlphaFold 3, which operates on the pair representation alone (no MSA dimension) using triangle attention and multiplicative updates.

Wigner D-matrix. The matrix representation $D_l(R)$ of a rotation $R \in SO(3)$ in the ℓ-th irrep. Entries are given by $D_l_mm'(\alpha, \beta, \gamma)$ where α, β, γ are the Euler angles of R. Wigner D-matrices satisfy the composition rule $D_l(R_1R_2) = D_l(R_1)D_l(R_2)$.

Appendix C: Technical Supplement for Part VIII — Agentic AI, Meta-Learning, and Continual Learning

This appendix provides extended derivations and definitions supporting Part VIII. All notation follows Appendix A.

C.1 POMDP Formalism: Extended Definitions and the Belief-State MDP

A Partially Observable Markov Decision Process (POMDP) is defined by the tuple (S, A, O, P, R, Ω, γ) introduced in Part VIII, Chapter 1. The key reduction to a belief-state MDP proceeds as follows.

Definition C.1 (Belief state). The belief state at time t is the posterior distribution $b_t \in \Delta(S)$ defined by $b_t(s) = P(S_t = s \mid o_1, a_1, \ldots, o_t)$. It is updated via Bayes' rule after each observation:

$$b'(s') = \Omega(o' \mid s', a) \, \Sigma_s P(s' \mid s, a) \, b(s) \; / \; P(o' \mid b, a)$$

This defines the belief update operator $\tau: \Delta(S) \times A \times O \to \Delta(S)$. The belief-state MDP has state space $\Delta(S)$ (a continuous simplex), action space A, and reward $\underline{R}(b, a) = \Sigma_s \, b(s) \, R(s, a)$. An optimal POMDP policy $\pi^*: \Delta(S) \to A$ maps belief states to actions and is equivalent to an optimal policy for the belief-state MDP.

Definition C.2 (Value function of a POMDP). The optimal value function $V^*: \Delta(S) \to \mathbb{R}$ satisfies the Bellman equation:

$$V^*(b) = \max_a \left[\underline{R}(b, a) + \gamma \, \Sigma_{o'} P(o' \mid b, a) \, V^*(\tau(b, a, o')) \right]$$

V^* is piecewise linear and convex in b (it is the upper envelope of a set of linear α-vectors). Exact POMDP solvers (PBVI, SARSOP) approximate V^* over a finite set of belief points. For LLM agents, the context window serves as a finite-dimensional approximation to b, with the attention mechanism implementing a soft version of the belief update operator τ.

C.2 EWC as Laplace Approximation: Complete Derivation

We derive the Elastic Weight Consolidation objective from the Bayesian perspective, filling in the steps sketched in Theorem 5.1 of Part VIII.

Starting point. After learning task $\mathcal{T}_1$, the posterior over parameters is $p(\theta \mid \mathcal{D}_1)$. When learning task $\mathcal{T}_2$, Bayes' rule gives:

$$\log p(\theta \mid \mathcal{D}_1, \mathcal{D}_2) = \log p(\mathcal{D}_2 \mid \theta) + \log p(\theta \mid \mathcal{D}_1) + \text{const}$$

Laplace approximation of $p(\theta \mid \mathcal{D}_1)$. At the MAP estimate $\theta_1^* = \arg\max p(\theta \mid \mathcal{D}_1)$, the second-order Taylor expansion gives:

$$\log p(\theta \mid \mathcal{D}_1) \approx \log p(\theta_1^* \mid \mathcal{D}_1) - (1/2)(\theta - \theta_1^*)^T H_1 (\theta - \theta_1^*)$$

where $H_1 = -\nabla^2_\theta \log p(\theta_1^* \mid \mathcal{D}_1)$ is the Hessian of the negative log-posterior at θ_1^*. Since $\log p(\theta \mid \mathcal{D}_1) = \log p(\mathcal{D}_1 \mid \theta) + \log p(\theta) - \text{const}$, and using the Fisher identity $F_1 \approx H_1$ (valid at a maximum likelihood point for a well-specified model), we obtain:

$$\log p(\theta \mid \mathcal{D}_1) \approx \text{const} - (1/2) \sum_i F_{1,ii} (\theta_i - \theta_{1,i}^*)^2$$

where the last step uses the diagonal approximation $F_1 \approx \text{diag}(F_{1,11}, \ldots, F_{1,pp})$. The MAP estimator for the joint posterior then minimises:

$$\mathcal{L}_{\text{EWC}}(\theta) = \mathcal{L}_{\mathcal{T}_2}(\theta) + (\lambda/2) \sum_i F_{1,ii} (\theta_i - \theta_{1,i}^*)^2$$

which is precisely the EWC objective. The diagonal Fisher $F_{1,ii} = \mathbb{E}_{p_{\text{data}}}[(\partial \log p_\theta(y \mid x) / \partial \theta_i)^2]$ at $\theta = \theta_1^*$ is computed on the task-1 training set as a sample average. ■

C.3 Topological Data Analysis: Key Definitions

This section provides precise definitions for the topological objects used in Part VIII, Chapter 6.

Definition C.3 (Simplicial complex). A simplicial complex K on a vertex set V is a collection of subsets (simplices) of V such that (i) every singleton $\{v\} \in K$, and (ii) if $\sigma \in K$ and $\tau \subseteq \sigma$, then $\tau \in K$. A k-simplex has $k+1$ vertices. The Vietoris-Rips complex $\text{VR}_\varepsilon(X)$ at scale ε is a simplicial complex on point cloud X where a subset $\sigma \subseteq X$ is a simplex iff $\text{diam}(\sigma) \le \varepsilon$.

Definition C.4 (Homology group). For a simplicial complex K with coefficient ring $\mathbb{Z}/2\mathbb{Z}$, the k-th homology group $H_k(K) = \ker(\partial_k) / \text{im}(\partial_{k+1})$ where ∂_k is the boundary operator mapping k-chains to $(k-1)$-chains. The rank $\beta_k = \text{rank}(H_k(K))$ is the k-th Betti number: β_0 = connected components, β_1 = independent loops, β_2 = enclosed voids.

Definition C.5 (Persistence diagram). Given a filtration $K_0 \subseteq K_1 \subseteq \ldots \subseteq K_n$, a homological feature (class $\alpha \in H_k$) is born at $\varepsilon_b = \min\{\varepsilon_i : \alpha \in \text{im}(H_k(K_i) \to H_k(K_n))\}$ and dies at $\varepsilon_d = \min\{\varepsilon_j > \varepsilon_b : \alpha \in \text{im}(H_k(K_b) \to H_k(K_j)) = 0\}$. The persistence of α is $\text{pers}(\alpha) = \varepsilon_d - \varepsilon_b$. The persistence diagram D_k is the multiset of birth-death pairs $\{(\varepsilon_b, \varepsilon_d)\}$ for all k-dimensional homological features.

Remark C.1 (Bottleneck distance and stability). The bottleneck distance $d_B(D, D') = \inf_\gamma \sup_{p \in D} \|p - \gamma(p)\|_\infty$ between persistence diagrams D, D' satisfies the stability theorem: if $f, g: X \to \mathbb{R}$ with $\|f - g\|_\infty \le \varepsilon$, then $d_B(D(f), D(g)) \le \varepsilon$. This guarantees that small perturbations of the loss landscape or network weights produce small changes in the persistence diagram, validating topological training diagnostics.

C.4 GRPO Objective: Detailed Derivation from PPO

GRPO (Group Relative Policy Optimisation) modifies PPO by replacing the learned value function baseline with a within-group reward normalisation. Here we trace the derivation carefully.

PPO baseline. PPO minimises the surrogate loss $\mathscr{L}_PPO = -\mathbb{E}[\min(r(\theta)\hat{A}, \text{clip}(r(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A})]$, where $r(\theta) = \pi_\theta(y|x) / \pi_\{\theta_old\}(y|x)$ is the probability ratio and $\hat{A} = (r - V_\varphi(x))$ is the advantage estimated using a learned value function V_φ.

GRPO substitution. For a problem x and G sampled responses $\{y_1, ..., y_G\}$ from $\pi_\{\theta_old\}$, the group-relative advantage replaces $V_\varphi(x)$ with the within-group mean reward:

$$\hat{A}_i = (r(x, y_i) - \text{mean_j } r(x, y_j)) / \text{std_j } r(x, y_j)$$

This is an unbiased estimator of the advantage: $\mathbb{E}_\{y\sim\pi\}[\hat{A}_i] = 0$ by the centring. The variance of $\hat{A}_i$ is $O(1/G)$: larger group size G reduces variance as more within-group comparisons are made. The KL penalty $\beta\, D_KL(\pi_\theta \,\|\, \pi_ref)$ prevents the policy from drifting too far from a reference (pre-RLHF) model, playing the same role as the KL term in RLHF with PPO. The elimination of V_φ saves approximately one forward pass and one backward pass per training step (the critic network), halving memory usage in practice.

Bibliography for Part VII: Scientific AI and Geometric Deep Learning

The following references are cited in Part VII. Works are listed alphabetically by first author's surname. All arxiv preprints were accessed and verified as of publication date of this treatise.

Primary Architecture Papers

Abramson, J., Adler, J., Dunger, J., et al. (2024). Accurate structure prediction of biomolecular interactions with AlphaFold 3. Nature, 630, 493–500. doi:10.1038/s41586-024-07487-w. Introduces AlphaFold 3, extending structure prediction to DNA, RNA, and small molecules via a diffusion module operating on 3D atom coordinates.

Benegas, G., Bhodiwal, S., Ye, C., & Song, Y. S. (2024). AlphaGenome: Advancing genomic sequence and variant effect prediction. bioRxiv. Presents a unified multi-task model for regulatory genomics, predicting RNA-seq, ATAC-seq, ChIP-seq, and Hi-C from raw DNA sequence using multi-scale convolution and reverse complement equivariance.

Hayes, T., Rao, R., Akin, H., et al. (2024). Simulating 500 million years of evolution with a language model. Science, 385(6707). Introduces ESM-3, a multimodal protein language model jointly modelling sequence, structure, and function through unified masked language modelling.

Jumper, J., Evans, R., Pritzel, A., et al. (2021). Highly accurate protein structure prediction with AlphaFold. Nature, 596, 583–589. doi:10.1038/s41586-021-03819-2. The definitive paper on AlphaFold 2; introduces the Evoformer, Invariant Point Attention, FAPE loss, and the dual MSA/pair representation.

Equivariant Networks and Geometric Deep Learning

Batzner, S., Musaelian, A., Sun, L., et al. (2022). E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. Nature Communications, 13, 2453. Introduces NequIP, a highly data-efficient interatomic potential using tensor-product-based SE(3)-equivariant message passing.

Batatia, I., Kovács, D. P., Simm, G., Ortner, C., & Csányi, G. (2022). MACE: Higher order equivariant message passing neural networks for fast and accurate force fields. NeurIPS 2022. Introduces MACE, which uses body-order truncation to make high-ℓ tensor products computationally feasible.

Cohen, T. S. & Welling, M. (2016). Group equivariant convolutional networks. ICML 2016. The foundational paper establishing group-equivariant CNNs; proves that convolution is the unique linear equivariant operation for locally compact groups.

Cohen, T. S., Geiger, M., Köhler, J., & Welling, M. (2018). Spherical CNNs. ICLR 2018. Extends equivariant CNNs to the sphere S^2 using the spherical harmonics as the relevant irreducible representations of SO(3).

Klicpera, J., Groß, J., & Günnemann, S. (2020). Directional message passing for molecular graphs. ICLR 2020. Introduces DimeNet, which incorporates bond angles as directional features, strictly increasing the expressivity of distance-only GNNs.

Liao, Y. L. & Smidt, T. (2023). Equiformer: Equivariant graph attention transformer for 3D atomistic graphs. ICLR 2023. Brings the Transformer architecture to SE(3)-equivariant settings via equivariant dot-product attention.

Schütt, K., Kindermans, P.-J., Felix, H. E. S., Chmiela, S., Tkatchenko, A., & Müller, K.-R. (2017). SchNet: A continuous-filter convolutional neural network for modeling quantum interactions. NeurIPS 2017. Introduces the first scalable distance-based GNN for molecular property prediction.

Schütt, K., Unke, O., & Gastegger, M. (2021). Equivariant message passing for the prediction of tensorial properties and molecular spectra. ICML 2021. Introduces PaiNN, which augments scalar features with equivariant vector features, enabling force prediction and chiral discrimination.

Satorras, V. G., Hoogeboom, E., & Welling, M. (2021). E(n) equivariant graph neural networks. ICML 2021. A computationally lightweight alternative to spherical-harmonic-based equivariant GNNs, achieving E(n)-equivariance without explicit irrep decompositions.

Thomas, N., Smidt, T., Kearnes, S., et al. (2018). Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds. arxiv:1802.08219. The first explicit use of spherical harmonics and Clebsch-Gordan products in equivariant neural networks for molecular data.

Group Theory and Representation Theory

Bronstein, M. M., Bruna, J., Cohen, T., & Veličković, P. (2021). Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. arxiv:2104.13478. The comprehensive mathematical treatment of geometric deep learning as a unification of existing architectures through the lens of symmetry and equivariance.

Fulton, W. & Harris, J. (1991). Representation Theory: A First Course. Springer Graduate Texts in Mathematics, vol. 129. The standard reference for the representation theory of Lie groups, including SO(3), SU(2), and their Clebsch-Gordan theory.

Weyl, H. (1946). The Classical Groups: Their Invariants and Representations (2nd ed.). Princeton University Press. The classical reference connecting group invariant theory to physics and geometry, underpinning the equivariance framework.

Multi-Task Learning and Gradient Methods

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., & Finn, C. (2020). Gradient surgery for multi-task learning. NeurIPS 2020. Introduces PCGrad, the gradient projection method for resolving conflicting gradients in multi-task settings, applied to regulatory genomics in AlphaGenome.

Bibliography for Part VIII: Agentic AI, Meta-Learning, and Continual Learning

The following references are cited in Part VIII. Works are listed alphabetically by first author's surname.

Agentic Systems and Multi-Agent Reinforcement Learning

Akyürek, E., Schuurmans, D., Andreas, J., Ma, T., & Zhou, D. (2023). What learning algorithm is in-context learning? Investigations with linear models. ICLR 2023. Proves that transformer in-context learning implements ridge regression via a gradient-descent-like mechanism in the attention layers.

Chaudhry, A., Ranzato, M. A., Rohrbach, M., & Elhoseiny, M. (2019). Efficient lifelong learning with A-GEM. ICLR 2019. Introduces A-GEM, a computationally efficient approximation to GEM that projects the gradient onto a single averaged past-task constraint.

Lasry, J.-M. & Lions, P.-L. (2007). Mean field games. Japanese Journal of Mathematics, 2(1), 229–260. The foundational paper establishing mean-field game theory and the forward-backward equation system for the N→∞ limit.

Lopez-Paz, D. & Ranzato, M. A. (2017). Gradient episodic memory for continual learning. NeurIPS 2017. Introduces GEM, which enforces non-negative inner products between the current gradient and stored past-task gradients as an exact non-forgetting constraint.

Rashid, T., Samvelyan, M., Schroeder de Witt, C., Farquhar, G., Foerster, J., & Whiteson, S. (2018). QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. ICML 2018. Introduces the QMIX mixing network and the Individual-Global-Max (IGM) condition for cooperative MARL.

Yao, S., Zhao, J., Yu, D., et al. (2022). ReAct: Synergizing reasoning and acting in language models. ICLR 2023. Introduces the ReAct framework for LLM agents, interleaving chain-of-thought reasoning with tool-call actions within the context window.

Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., & Wu, Y. (2022). The surprising effectiveness of PPO in cooperative multi-agent games. NeurIPS 2022. Demonstrates that MAPPO with centralised critics is competitive with specialised MARL algorithms on StarCraft and other benchmarks.

Inference-Time Compute and Reinforcement Learning from Human Feedback

Coulom, R. (2006). Efficient selectivity and backup operators in Monte-Carlo tree search. Computers and Games, LNCS 4630. The paper introducing MCTS with the UCB1 selection rule; foundational for AlphaGo and subsequent work on MCTS for LLMs.

Lightman, H., Kosaraju, V., Burda, Y., et al. (2023). Let's verify step by step. arxiv:2305.20050. Introduces Process Reward Models trained on step-by-step human annotations,

demonstrating their superiority over outcome-only reward models for mathematical reasoning.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal Policy Optimization algorithms. arxiv:1707.06347. The PPO paper; introduces the clipped surrogate objective and the KL-penalty variant, both used in GRPO.

Shao, Z., Wang, P., Zhu, Q., et al. (2024). DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. arxiv:2402.03300. Introduces GRPO (Group Relative Policy Optimisation) and demonstrates its effectiveness for mathematical reasoning without a critic network.

Snell, C., Lee, J., Xu, K., & Kumar, A. (2024). Scaling LLM test-time compute optimally. arxiv:2408.03314. A systematic study of test-time compute scaling strategies (Best-of-N, beam search, PRM-guided MCTS), establishing the optimal allocation of inference-time compute.

Video Generation and Diffusion Models

Brooks, T., Peebles, B., Holmes, C., et al. (2024). Video generation models as world simulators. OpenAI Technical Report. Introduces Sora: a video generation model based on DiT with spatiotemporal patch tokenisation and 3D RoPE, trained to produce physically coherent videos of arbitrary resolution and duration.

Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., & Fleet, D. J. (2022). Video Diffusion Models. NeurIPS 2022. The foundational paper on diffusion models for video generation; introduces factored spatial and temporal attention to reduce the $O((THW)^2)$ computational cost of full spatiotemporal attention.

Yang, Z., Teng, J., Zheng, W., et al. (2024). CogVideoX: Text-to-video diffusion models with an expert transformer. arxiv:2408.06072. Demonstrates high-quality text-to-video synthesis using 3D full attention with flow matching in approximately 50 sampling steps.

Meta-Learning

Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. ICML 2017. Introduces MAML; proves that a single gradient step on a small support set, followed by a meta-gradient through this step, produces fast task adaptation across diverse problem domains.

Nichol, A., Achiam, J., & Schulman, J. (2018). On first-order meta-learning algorithms. arxiv:1803.02999. Introduces Reptile and FOMAML; proves that Reptile implicitly maximises within-task gradient alignment, providing a first-order justification for MAML-like meta-learning.

Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. NeurIPS 2017. Introduces ProtoNets, where class prototypes are the means of support-set embeddings and classification uses nearest-prototype in Euclidean space.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., & Wierstra, D. (2016). Matching networks for one shot learning. NeurIPS 2016. The seminal few-shot learning paper

introducing attention-based matching over a support set, predating and motivating subsequent Transformer-based approaches.

Continual Learning

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., et al. (2017). Overcoming catastrophic forgetting in neural networks. Proceedings of the National Academy of Sciences, 114(13), 3521–3526. Introduces Elastic Weight Consolidation (EWC), using the Fisher information matrix diagonal as parameter importance weights in an L2-regularisation term that penalises forgetting.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., et al. (2016). Progressive neural networks. arxiv:1606.04671. Introduces Progressive Neural Networks, which add a new column of parameters per task with lateral connections to previous frozen columns, guaranteeing zero forgetting by construction.

Zenke, F., Poole, B., & Ganguli, S. (2017). Continual learning through synaptic intelligence. ICML 2017. Introduces Synaptic Intelligence (SI), an online variant of EWC that accumulates parameter importance weights during training based on each parameter's contribution to the loss decrease.

Topological Analysis of Learning

Carlsson, G. (2009). Topology and data. Bulletin of the American Mathematical Society, 46(2), 255–308. The foundational survey of persistent homology and topological data analysis; introduces the persistence diagram and the stability theorem.

Cohen-Steiner, D., Edelsbrunner, H., & Harer, J. (2007). Stability of persistence diagrams. Discrete and Computational Geometry, 37(1), 103–120. Proves the bottleneck stability theorem (Theorem 6.1 of Part VIII), establishing that persistence diagrams vary continuously with the underlying function.

Frankle, J. & Carlin, M. (2019). The lottery ticket hypothesis: Finding sparse, trainable neural networks. ICLR 2019. Empirically identifies sparse subnetworks ("winning tickets") that match the performance of dense networks when trained in isolation, motivating the study of loss landscape geometry.

Nanda, N., Chan, L., Lieberum, T., Smith, J., & Steinhardt, J. (2023). Progress measures for grokking via mechanistic interpretability. ICLR 2023. Provides the mechanistic account of grokking for modular arithmetic; identifies the Fourier-circular representation and the phase transition in H_1 persistent homology as the mechanism of delayed generalisation.

Naitzat, G., Zhitnikov, A., & Lim, L.-H. (2020). Topology of deep neural networks. Journal of Machine Learning Research, 21(184), 1–40. Analyses the topological transformation of data representations across layers, showing that deep networks progressively reduce the Betti numbers of class boundaries.